Eötvös Loránd Tudományegyetem

Társadalomtudományi Kar

MESTERKÉPZÉS

A BERTopic alkalmazásának lehetőségei és korlátai

Konzulens:
Rakovics Zsófia

Készítette:
Piros Anna Sára
N4IBUA
Survey statisztika és adatanalitika szak

2024. április

TARTALOMJEGYZÉK

KÖSZÖNETNYILVÁNÍTÁS

Ezúton szeretnék köszönetet mondani az ELTE Research Center for Computational Social Science kutatócsoportnak és Rakovics Zsófiának, a kutatócsoport kutatójának, amiért elérhetővé tették számomra Orbán Viktor miniszterelnöki beszédeinek angol nyelvű korpuszát. Külön szeretnék köszönetet mondani Rakovics Zsófiának támogatásáért és tanácsaiért, amivel végigkísérte a munkámat.

ABSZTRAKT

Egy új topikmodellezési technika, a BERTopic működését és teljesítményét mutatom be az elterjed LDA modellel szemben. A gyakorlati összehasonlításhoz egy LDA és két BERTopic modellt vizsgáltam Orbán Viktor angol nyelvű miniszterelnöki beszédeinek korpuszán. Az optimalizált LDA modellnél meghatározott beállításokat alkalmaztam az egyik BERTopic modellen, és optimalizált beállításokat a másikon. A modellek kiértékeléséhez topikkoherencia és topikdiverzitás mutatókat, valamint a topikreprezentációk értelmezhetőségét vizsgáltam. Az optimalizált LDA modell redundáns és nem összefüggő topikokat eredményezett, míg mindkét BERTopic modell változatos, koherens és specifikus topikokat hozott létre. A BERTopic jobb eredményeket ér el, alkalmazása egyszerűbb és számos lehetőség rejlik benne a moduláris, flexibilis felépítésének köszönhetően.

Kulcsszavak: BERTopic, LDA, NLP, topikmodell, topikszám, stopszavak, SBERT, UMAP, HDBSCAN, c-TF-IDF, Python

Legfontosabb szakirodalmak:

Grootendorst, M. (2022): BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794*., 1-10.

Egger, R. – Yu J. (2022): A Topic Modeling Comparison Between LDA, NMF, Top2Vec, and BERTopic to Demystify Twitter Posts. *Frontiers in sociology*, 7(886498): 1-16.

BEVEZETÉS

A természetesnyelv-feldolgozás területén napról napra újabb és hatékonyabb módszerek jelennek meg a szakirodalomban, melyek segítségével a szövegek által hordozott információkat tudjuk kinyerni és strukturálni. A hosszú szöveges dokumentumokban lévő látens témakörök feltárására alkalmas módszerek közül az egyik legújabb és egyben legígéretesebb a BERTopic eljárás, melyet először 2022-ben mutatott be Grootendorst a szakirodalomban. A BERTopic egy olyan topikmodellezési módszer, amelyben moduláris felépítésének köszönhetően rugalmasan cserélhetőek az egyes lépéseihez használt eszközök, így a széles körben elterjedt és a felsőoktatásban tanított topikmodellező technikáknál jobb teljesítményt ígér, amely lépést tud tartani a természetesnyelv-feldolgozás technikai fejlődésével.

Dolgozatomban bemutatom a BERTopic működését, gyakorlati relevanciáját és szükségességének indoklását. Ehhez összehasonlítom az elterjedt és népszerű LDA (Latent Dirichlet Allocation) modellel, kiemelve a hozzá viszonyított előnyeit és hátrányait. Az összehasonlításhoz használt korpuszom Orbán Viktor 2011 és 2023 közötti angol nyelvű miniszterelnöki beszédeiből áll. A gyakorlati alkalmazás valós korpuszon lehetővé teszi a modellek jóságának objektív összehasonlítását mind kvantitatív, mind pedig kvalitatív szempontok alapján. A kvantitatív szempontok között kiemelt figyelmet szentelek a topikkoherenciát és topikdiverzitást mérő mutatóknak, melyek segítségével megállapíthatjuk a modellek által generált topikok összetettségét és szemantikai egységét. Emellett a kvalitatív elemzésben a topikok értelmezhetőségét és a modellek használatának praktikusságát is vizsgálom. Az összehasonlítás során részletesen kitérek a BERTopic korlátaira és további lehetőségeire, melyek alapján lehetőséget látok a jelenlegi topikmodellező módszerekkel szembeni előnyeinek kiaknázására.

Fontos célja a dolgozatomnak, hogy útmutatóként is szolgáljon a BERTopic alkalmazásához. Emiatt részletesen tárgyalom a BERTopic paramétereinek beállítását, optimalizálási lehetőségeit és a modell használata közben felmerülő lehetséges akadályokat, valamint azok megoldását. Ennek érdekében áttekintem a BERTopic konfigurációs opcióit, és gyakorlati példákon keresztül mutatom be a modell használatának lehetőségeit és korlátait. A célom az,

hogy a dolgozat ne csupán elméleti megközelítést nyújtson a BERTopic-ról, hanem segítséget és iránymutatást is adjon annak gyakorlati alkalmazásához és esetleges problémáinak megoldásához.

Dolgozatom első fejezetében átfogóan bemutatom a vizsgált modellek elméleti hátterét, szakirodalmát. Bemutatom a természetesnyelv-feldolgozás alapfogalmait és főbb feladatait, majd kifejtem a topikmodellezés lényegét és céljait. Ismertetem az LDA (Latent Dirichlet Allocation) modellt, mint az egyik legelterjedtebb és legnépszerűbb topikmodellező módszert, kiemelve annak működési elvét és alkalmazási területeit. A fejezet fő fókuszában a BERTopic modell bemutatása áll, részletesen tárgyalom a működésének lépéseit és elméleti hátterét, valamint ismertetem a modellre vonatkozó tanulmányok eredményét, amelyek bemutatják az előnyeit és hátrányait más topikmodellekkel szemben.

A második fejezetben részletesen ismertetem a kutatás módszertanát. Bemutatom a kutatáshoz használt korpusz jellemzőit, valamint az LDA és BERTopic modellek Python implementációját. Különös hangsúlyt fektetek a paramétereik finomhangolási és optimalizálási lehetőségeinek bemutatására, a modellépítés folyamatának ismertetésére. A kutatásom során két BERTopic modellt építek, az egyiket az optimalizált LDA modell beállításaival paraméterezem, hogy megkapjam a tisztán a modellből adódó teljesítménybeli eltéréseket, a másik BERTopic modellben pedig a lehető legoptimálisabb beállításokat szeretném elérni, hogy feltárjam a modell lehetőségeit.

A harmadik fejezetben prezentálom a vizsgált modellek eredményeit. Ehhez részletesen ismertetem a Python implementációk által nyújtott vizualizációs lehetőségeket, melyek segítségével áttekinthetőek a generált topikok és azok jellemzői. Az eredmények összehasonlítása során részletesen elemzem és összevetem az egyes modellek teljesítményét. Különös figyelmet fordítok a generált topikok minőségére, a topikkoherenciára és topikdiverzitásra, valamint a modellek általános használhatóságára és praktikusságára. A fejezet célja, hogy gyakorlati alkalmazáson keresztül mutassam be a modellek lehetőségeit és korlátait.

1. ELMÉLETI ÁTTEKINTÉS

A kutatásom fontos lépése az elméleti áttekintés, a vonatkozó hazai és nemzetközi szakirodalom bemutatása. A fejezet fő célja a BERTopic működésének elméleti ismertetése, a lépéseinek részletes bemutatása és a teljesítményét elemző tanulmányok eredményének tárgyalása. Ehhez szükséges ismerni a természetes-nyelvfeldolgozás területét, így először átfogóan bemutatom a terület alapfogalmait és főbb feladatait, valamint a topikmodellezés lényegét és céljait. Ismertetem az LDA modell működését és lehetőségeit is, hiszen az összehasonlítás során fontos értenünk a számításaik eltérő metódusát, illetve ismernünk az alternatív lehetőségeket.

1.1. TERMÉSZETESNYELV-FELDOLGOZÁS (NLP)

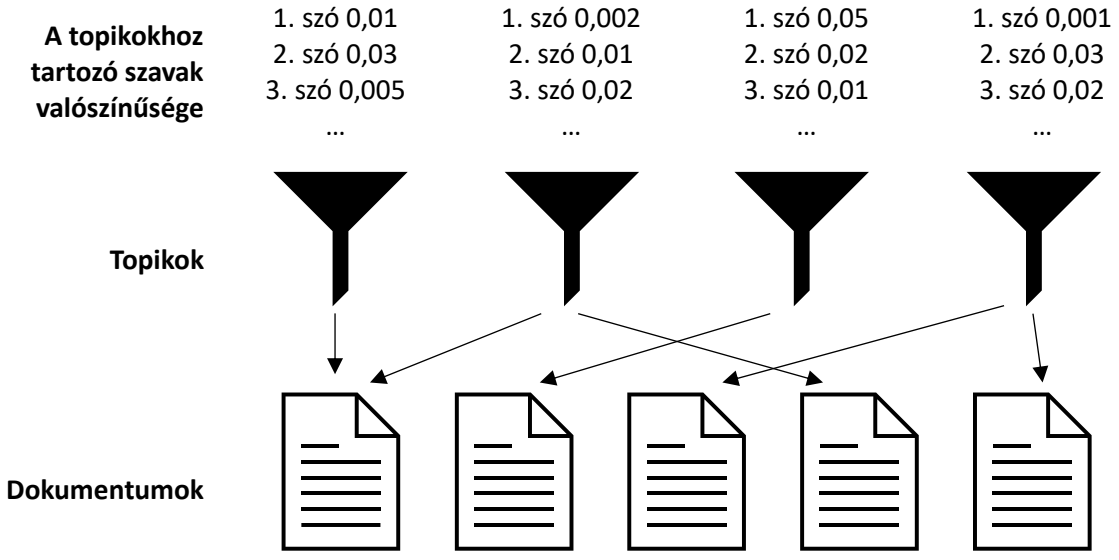
A Natural Language Processing (rövidítve NLP, természetesnyelv-feldolgozás és természetes nyelvfeldolgozás írásmóddal is használja a magyar szakirodalom) olyan módszereket foglal magába, amelyek természetes nyelven[1] előállt szövegek nagy mennyiségét képesek értelmezni, elemezni és akár előállítani is (Hirschberg–Manning, 2015) az informatika, mesterségesintelligencia-kutatás és nyelvészet ötvözésével (Németh et al., 2020). Az NLP az utóbbi évtizedekben egyre növekedő népszerűségnek örvend a tudományos életben és az ipari alkalmazásban is (pl. beszédfelismerő szoftverek elterjedése). Ezt a fellendülést elősegítette a nagy mértékben megnövekedett számítási teljesítmény, a hatalmas mennyiségű szöveges adathoz való könnyű hozzáférés az internetnek köszönhetően (közösségi média, híroldalak, blogok), és mind a gépi tanulás módszereinek, mind a természetes nyelv szerkezetének és társadalmi kontextusba ágyazódásának megértésével kapcsolatos ugrásszerű fejlődés is. Az NLP legjelentősebb korlátja a nyelvi konfiguráció kihívása. Azok a nyelvek vannak előnyben, amelyek nagy mennyiségű szöveges adatforrással vannak jelen az interneten (high-resource languages, pl. angol, spanyol, kínai), a kevesebbek által beszélt és írt nyelvekre vagy nem

[1] "Természetes nyelvnek azokat a nyelveket nevezzük, amelyek spontán alakultak ki, amelyek nyelvtana az emberek közötti nyelvi kommunikáció természetes fejlődésének eredményeként alakult ki, ilyenek például a különböző nemzetek nyelvei. Ezzel szemben a mesterséges nyelvet az ember hozza létre, szabályai tudatosan tervezettek, ilyenek lehetnek a programozási nyelvek, de akár a morze is ide tartozhat." – Németh et al., 2020, p. 47.

készítik el a nyelvmodellek variációit, vagy azok később kerülnek megjelenésre (Hirschberg–Manning, 2015).

Az NLP módszereken belül léteznek felügyelt (supervised) és nem felügyelt (unsupervised) megközelítések. Az előbbinél rendelkezünk előzetesen ismert kategóriákkal és egy felcímkézett adathalmazzal, amin tanulhat a modellünk, hogy felismerje az egyes kategóriákon belüli szemantikai hasonlóságokat, valamint a kategóriák közötti különbözőségeket. A modell fejlesztésének a célja, hogy a tanulás után minél nagyobb pontossággal legyen képes a felcímkézetlen szövegeket a megfelelő kategóriákhoz rendelni. Fontos kitétel itt a túlillesztés elkerülése, annak érdekében, hogy más korpuszokon is jól működjön a modellünk. A nem felügyelt módszerek célja, hogy előzetesen nem ismert kategóriákat fedezzünk fel a korpuszban. Klasszifikációs problémákra és látens tartalmak kinyerésére is alkalmazhatjuk ezeket a módszereket, mint például a szóbeágyazási modellt, klaszterelemzést és a topikmodellezést (Németh et al., 2020).

A topikmodellek célja a dokumentumgyűjtemények látens témáinak azonosítása. Ellenben a klaszterelemzéssel, ahol az egyes szövegek elkülönített csoportokba (klaszterekbe) tartoznak, a topikmodellezésben a dokumentumok néhány téma (topik) keverékeként állnak elő (*1. ábra)*.



1. ábra: Dokumentumok létrejötte a topikmodell feltevései szerint
Forrás: Németh et al., 2020

Ezek a látens topikok generálják a dokumentumokat a hozzájuk tartozó szó-valószínűségeloszlás alapján (például egy gazdasági topikban az „inflációnak” nagyobb lesz a valószínűsége, mint egy művészeti topikban). Nem felügyelt módszerről van szó, tehát a topikok tartalma és számossága előzetesen nem ismert. Ugyanakkor a modell bemeneti paramétereként meg kell adni a topikok számát (K), és többféle K mellett illesztett modell közül választjuk ki a legjobbat, ezáltal a megfelelő topikszámot is. A topikmodellek interpretációjában a megtalált témák számát, népszerűségét, a hozzájuk tartozó legvalószínűbb szavakat és a topikok keveredési hajlandóságát is értelmezzük (Blei–Lafferty 2009-es könyvét hivatkozza Németh et al., 2020).

1.2. LATENT DIRICHLET ALLOCATION (LDA)

A leggyakrabban használt topikmodellezési technika az LDA (Latent Dirichlet Allocation, magyarul látens Dirichlet-allokáció) amelyet David Blei, Andrew Ng és Michael I. Jordan mutatott be 2003-ban. Az LDA egy háromszintű hierarchikus Bayesi modell, ahol minden dokumentumra témák (topikok) véges keverékeként tekintünk, és minden téma a mögöttes témavalószínűségek végtelen keverékeként van modellezve.

A modellnek három paramétere van: K, amely a topikok száma, alfa, mely a topikdiverzitást határozza meg, és béta, melynek nagysága a topikokhoz tartozó szószám nagyságát jelöli (Egger–Yu, 2022). Az egyes topikok a hozzájuk tartozó szavak eloszlását reprezentálják, amit a modell a dokumentumokban lévő szavak alapján próbál meg azonosítani. Az LDA modell a dokumentumok topikok szerinti eloszlását, és a topikok szavak szerinti eloszlását Dirichlet-eloszlásként kezeli, úgy hangolva azt, hogy minimalizálja a témák keveredését. Ez a feltételezés segíti a modellt abban, hogy rugalmasan kezelje a témák közötti különbözőségeket a dokumentumokban, lehetővé téve, hogy minden dokumentum különböző arányban tartalmazhasson témákat, és minden téma különböző szavakkal bővülhessen. Az LDA generatív valószínűségi modell, tehát a működése során minden dokumentumhoz és minden szóhoz rejtett változókat rendel, amelyek a topikokat jelölik. Az inferencia folyamatában a modell a dokumentumok szavainak eloszlását használja fel arra, hogy következtetéseket vonjon le a témák eloszlására vonatkozóan minden egyes dokumentumban. Az LDA alkalmazza az empirikus Bayesi paraméterbecslés Expectation-Maximization (EM) algoritmusát a témák és a

hozzájuk tartozó szavak valószínűségi eloszlásának meghatározására. Az LDA sajátossága, hogy alapvetően minden dokumentumban megtalálhatóak ugyanazok a topikok, eltérő arányban keveredve egymással, így leszűkíthetjük az egyes dokumentumokat a rájuk legjellemzőbb topikok keverékére (Blei et al., 2003; Blei, 2012).

Blei 2012-es tanulmányában, közel tíz évvel az LDA bevezetése után bemutatja a modell továbbfejlesztéseit, amelyeket a klasszikus LDA különböző korlátait, feltételezéseit oldják fel. Az első ilyen megszorítás az LDA modellben, hogy a dokumentumokra „szózsákként" (angolul „bag of words") tekint, tehát nem veszi figyelembe a szavak sorrendjét. A kontextus elhagyása információveszteséggel járhat, így Wallach (2006) topikmodellje n-grammok (egymás után következő n darab szó láncolatai) összességeként értelmezi a dokumentumokat, így a szavak alapján azonosított témák a kontextust is figyelembe tudják venni. További előnye a fejlesztésnek, hogy az eredeti LDA-tól eltérően nem a funkciószavak dominálják a felfedezett topikokat. A következő korlátja a modellnek, hogy a dokumentumok sorrendjét sem veszi figyelembe, pedig bizonyos esetekben a témastruktúra időbeli változásának feltárása kardinális kérdése a kutatásnak (például, ha azt szeretnénk vizsgálni, hogy egy híroldal cikkeinek látens témái hogyan követték le a politikai eseményeket), ennek feloldására a dinamikus topikmodellt vezette be Blei és Lafferty (2006). A szekvenciális topikmodelljük több, időben egymást követő eloszlást határoz meg az eredeti, témánkénti egy eloszlás helyett, Gauss-idősorok, Kálmán-szűrők és wavelet regresszió alkalmazásával. A harmadik korlátja az LDA modellnek, hogy a topikok számát ismertnek és fixáltnak értelmezi (ahogyan korábbiakban is olvasható, több különböző modellt állítunk fel eltérő topikszámokkal, és ezek jóságának összehasonlításával határozzuk meg az optimális topikszámot). Ennek a feloldására a HDP-t (hierarchical Dirichlet process, magyarul hierarchikus Dirichlet folyamat) vezették be, amely egy nemparaméteres Bayesi modell. A módszer a topikok számát a korpusz alapján határozza meg, és megengedi a számuk dinamikus változását újabb dokumentumok hozzáadásával. A modell értelmezhető az eloszlások eloszlásaként (alapértékük alapján csoportosított Dirichlet-folyamatok elosztása egy másik Dirichlet-folyamat szerint), erre utal a hierarchia a megnevezésében (Tah et al., 2004; Tah et al., 2006). A számos fejlesztés és további topikmodellező technikák megjelenése ellenére a klasszikus LDA a mai napig a legelterjedtebb topikmodellezési módszer, amelyet a használatának egyszerűsége indokol (Egger-Yu, 2022).

Számos magyar kutatás is alkalmazza az LDA módszert, főleg társadalomtudományi és politikai témájú elemzésekre. Az LDA-t alkalmazó magyar kutatások közül néhányat az *1. táblázatban* ismertetek.

Szerző(k)	Évszám	Tanulmány címe
Balogh Kitti	2015	A látens Dirichlet allokáció társadalomtudományi alkalmazása - A kuruc.info romaellenes megnyilvánulásainak tematikus elemzése*
Barna Ildikó, Knap Árpád	2019	Antisemitism in Contemporary Hungary: Exploring Topics of Antisemitism in the Far-Right Media Using Natural Language Processing
Tóbiás Dániel	2020	A nemi diszkrimináció megjelenésének elemzése Twitch.tv csatornákon szövegbányászati módszerek segítségével*
Katona Eszter, Kmetty Zoltán, Németh Renáta	2021	A korrupció hazai online médiareprezentációjának vizsgálata természetes nyelvfeldolgozással
Máté Fanni, Katona Eszter, Knap Árpád, Csótó Mihály	2021	Az Információs Társadalomban megjelenő tanulmányok topikelemzése
Knap Árpád, Bartha Diána, Barna Ildikó	2021	Trianon és a holokauszt emlékezetpolitikai jellegzetességeinek elemzése természetes-nyelv-feldolgozás használatával
Zaboretzky Bendegúz	2021	Depresszió és COVID-19 – online fórumok topik modellezése*
Berbekár Réka	2022	Trianon emlékezetpolitikájának vizsgálata gépi tanulási és szöveganalitikai eszközökkel*
Gelányi Péter, Sebők Miklós, Ring Orsolya	2022	A topikmodellezés lehetőségei és korlátai egy törvénykorpusz példáján

* A Survey statisztika és adatanalitika mesterképzésen íródott szakdolgozat

1. táblázat: Magyar kutatások az LDA használatával

1.3. BERTOPIC

A BERTopic modellt Grootendorst vezette be 2022-ben. A széles körben használt LDA és NMF topikmodelleknek azt a korlátját oldja fel, hogy szózsák modellekként értelmezik a dokumentumokat, figyelmen kívül hagyva a szavak kontextusát. Ezen túl a moduláris felépítésének köszönhetően rugalmasan cserélhetőek az egyes lépéseihez használt eszközök, így a széles körben elterjedt technikákkal ellentétben lépést tud tartani a természetesnyelv-feldolgozás technikai fejlődésével.

A BERTopic modell a klaszterező technikák és a TF-IDF (term frequency-inverse document frequency) osztályalapú variációjának az ötvözése. Szövegbeágyazásra és klaszterezésre épülő topikmodellezési technika a Sia és társai által 2020-ban bemutatott modell, és az Angelov által szintén 2020-ban bevezetett Top2Vec modell is, azzal az eltéréssel, hogy ezek centroid-alapú megközelítést alkalmaznak. Abból indulnak ki, hogy a beágyazás minél közelebb van a klaszter vagy topik súlypontjához, annál jobban jellemzi azt. Grootendorst elveti ezt a megközelítést arra alapozva, hogy a klaszterhez tartozó elemek nem feltétlenül a súlypont körüli kör alakzatban helyezkednek el.

A BERTopic modell topikképzési technikája három részre tagolható:

I. Dokumentumok vektortérbeli reprezentációjának elkészítése szövegbeágyazással
II. Dimenziószám csökkentése és klaszterek létrehozása
III. Klaszterek topikhoz rendelése

1.3.1. Dokumentumok beágyazása

A modell először minden dokumentumnak elkészíti a vektortérbeli reprezentációját alapértelmezetten az SBERT szövegbeágyazási technikával, de a BERTopicon belül használt szövegbeágyazási technika szabadon választható, azzal a feltétellel, hogy megfelelően reprezentálja a szövegek szemantikai hasonlóságát. Ez hatalmas előnye a modellnek, hiszen ez a rugalmasság biztosítja, hogy a modell nem avul el, a szövegbeágyazási technikák fejlődésével együtt tud haladni. A szó- vagy mondatbeágyazási technikák neurális hálókat használó technikák, a korpusz szemantikai struktúráját reprezentálják vektortérben. Egy vektor egy szónak (vagy szövegrésznek) felel meg, az elhelyezkedésüket pedig a jelentésük, pontosabban a használatuknak szövegkörnyezete határozza meg. A vektortérben lévő távolságukat az szabja

meg, hogy milyen gyakran fordulnak elő egymás közelében a korpuszban (Németh et al., 2020). A BERT (Devlin et al., 2018) újítása a többi nyelvmodellhez képest, hogy a pre-training során a jobb és baloldali kontextust is figyelembe veszi a reprezentációk létrehozásakor. A SBERT (Reimers-Gurevych, 2019) ennek a modellnek egy variációja, amely sziámi és triplet hálózatok használatával képes szemantikai hasonlóságra épülő, rögzített méretű mondatbeágyazásokat létrehozni. A sziámi hálózat két azonos alhálózatot tartalmaz, amelyek egy-egy bemenet vektorrá alakítását végzik el párhuzamosan, majd a kimeneti értékek összehasonlításával dönt az osztályozó lépés során, hogy azonosnak tekinthetőek-e a bemenetek. Alkalmazható például képfelismerésre, kézírás tulajdonosának azonosítására is (Bromley et al., 1994). Az SBERT a sziámi hálózatot a mondatok szemantikai hasonlóságának meghatározására használja. Az alhálózatai azonos BERT modellek, amelyek u és v vektorrá alakítják a bemeneti mondatokat a szavak vektorreprezentációinak összevonásával. Ezután különböző célfüggvények alkalmazhatóak, például koszinusz hasonlóság[2] számítása (regressziós célfüggvény), vagy osztályozás. Az osztályozási célfüggvényben összevonásra kerül az u és v vektor az $|u - v|$ elemenkénti különbséggel, majd a $W_t \in \mathbb{R}^{3n \times k}$ súllyal szorzódik. Ennek vesszük a normalizált exponenciális függvényét (más néven softmax függvény), ami a logisztikus függvény általánosítása több dimenzióra:

$$o = softmax(W_t(u, v, |u - v|))$$

A triplet hálózat értelemszerűen három azonos alhálózatot tartalmaz. Az SBERT triplet célfüggvényében bemenetként három mondat kerül megadásra, egy pozitív p mondat, egy negatív n mondat, és az a referenciamondat („anchor”). A triplet veszteség minimalizálása úgy hangolja a hálózatot, hogy az a és p között kisebb legyen a távolság, mint az a és n között:

$$max(\|s_a - s_p\| - \|s_a - s_n\| + \epsilon, 0)$$

Az s a mondatok beágyazását jelöli, a $\|\cdot\|$ a távolságot és a ϵ a margót (általában 1), ami biztosítja, hogy legalább ennyivel közelebb legyen s_p és s_a, mint s_n és s_a (Reimers-Gurevych, 2019, p. 3984.).

[2] A koszinusz hasonlóság a két vektor által bezárt szög koszinusza.

1.3.2. Dimenziószám csökkentése és klaszterek létrehozása

A következő lépés a dokumentumbeágyazás folyamán nagy mértékben megnövekedett dimenziószámot kezeli, amely megnehezíti a vektortérbeli elhelyezkedés és távolságok értelmezhetőségét. A klaszterelemzés ugyan önmagában is dimenziócsökkentő eljárás, de az előzetes csökkentése a dimenziószámnak UMAP (Uniform Manifold Approximation and Projection) eljárással optimalizálja a klaszterezés folyamatát. A UMAP jól teljesít a lokális és globális jellemzők megőrzésében a dimenziócsökkentés során, valamint kevés alkalmazási korlátja miatt rugalmasan használható. Az eljárás első lépésben Riemann[3] metrikahasználatával különböző távolságokat rendel a pontokhoz, hogy azok egyenletes eloszlását biztosítsa a sokaságon (manifold). A létrejött lokális adatpontok a különböző távolságok miatt azonban inkompatibilisek lehetnek, így azokat a módszer egyesíti és átalakítja fuzzy topológiai reprezentációvá. Az $X = \{X_1, \dots, X_2\} \in \mathbb{R}^n$ adathalmaz reprezentációjának létrehozásához az $\{(X, d_i)\}_{i=1\dots N}$ kiterjesztett pszeudo-metrikus terek családját használja, ahol az X a közös halmaz (common carrier set):

$$d_i(X_j, X_k) = \begin{cases} d_{\mathcal{M}}(X_j, X_k) - p, & ha\ i = j\ vagy\ i = k \\ \infty, & minden\ más\ esetben \end{cases}$$

A p-nek az X_i legközelebbi szomszédos ponttól (nearest neighbor) vett távolsága, $d_{\mathcal{M}}$ pedig az a priori geodézikus távolság az M sokaságon. Az X fuzzy reprezentációja a $FinSing$ fuzzy szinguláris halmaz funktorral kerül kiszámításra (McInnes et al., 2018, p. 9-11.):

$$\bigcup_{i=1}^{n} FinSing((X, d_i))$$

A csökkentett dimenziószámú adatpontokra is elvégzi a reprezentáció képzését, és az optimalizálás során a pontok átmozgatásával minimalizálja a két fuzzy reprezentáció ((A, μ) és (A, v)) C kereszt-entrópiáját, amely a következőképpen kerül kiszámításra (McInnes et al., 2018, p. 12.):

$$C\big((A, \mu), (A, v)\big) \triangleq \sum_{a \in A} \left(\mu(a) \log\left(\frac{\mu(a)}{v(a)}\right) + \big(1 - \mu(a)\big) log\left(\frac{1 - \mu(a)}{1 - v(a)}\right)\right)$$

[3] A Riemann-geometria az euklideszi geometria „görbült” változata, amelybe nem vezethető be egyenesvonalú koordináta-rendszer (Jánossy-Tasnádi, 2016)

A csökkentett dimenziószámú dokumentumbeágyazásokon alkalmazza a modell a HDBSCAN (Hierarchical density based clustering) hierarchikus klaszterelemző eljárást, amely képes változó eloszláson klasztereket találni és a paraméter választása robusztusabb a DBSCAN-nél (McInnes et al., 2017). Első lépésben meghatározza a sűrűséget a k. legközelebbi szomszédos ponttól vett távolság alapján ($core_k$ távolság). Az alacsony sűrűségű pontok vagy zaj megtalálásához a módszer az úgynevezett mutual reachability distance-t (továbbiakban mreach) számítja ki, ahol a $d(a,b)$ az eredeti távolság a két pont között (Asyaky-Mandala, 2021, p. 3.):

$$d_{mreach-k}(a,b) = max\{core_k(a), core_k(b), d(a,b)\}$$

A következő lépésben az mreach felhasználásával határozza meg a minimális élsúlyú feszítőfát[4] (minimum spanning tree) a sűrű területen lévő összefüggő pontok megtalálásához. Majd a fa „metszése" (pruning) történik meg, összehasonlítja az „ágaiban" (branch) lévő pontok számát a minimális klaszterméretttel. Ezután a „megmetszett" feszítőfa klaszterei stabilitásának kalkulációja történik, ahol a p adatpontok klaszterből való kiesésekori küszöbérték a λ_p, és a klaszterszétváláskori küszöbérték a λ_{birth} (Asyaky-Mandala, 2021, p. 3.):

$$\sum_{p \in klaszter} (\lambda_p - \lambda_{birth})$$

A stabilitás alapján kerülnek kiválasztásra a végső klaszterek. A minimális klaszterméret, ami alapján a „metszés" történik, változtatható. Ha egy klaszter nem tartalmaz ennek megfelelően elég adatpontot, akkor a zajhoz adódik, tehát azok az adatpontok outlierként lesznek megjelenítve.

1.3.3. Topikok létrehozása

A BERTopic utolsó lépésben minden klasztert egy topikhoz rendel, a klaszterekben lévő dokumentumok alapján modellezve a látens topikokat. Az osztály-alapúra módosított TF-IDF technikával állapítja meg a modell, hogy milyen fontos egy kifejezés az adott topikban, így határozva meg, hogy milyen témát fednek le az egyes topikok. Az eredeti TF-IDF módszer (Joachims 1996-os tanulmányát hivatkozza Grootendorst, 2022) a *t* kifejezés *d*

[4] A minimális élsúlyú feszítőfa a gráf olyan feszítőfája (a gráf minden csúcsát tartalmazó részgráfja), amelyben az élek összsúlya minimális (Temesi-Varró, 2017).

dokumentumbeli gyakoriságát ($tf_{t,d}$) megszorozza az inverz dokumentum gyakorisággal ($log\left(\frac{N}{df_t}\right)$). Utóbbi azt méri, hogy a *t* kifejezés mennyi információt szolgáltat egy dokumentum számára, úgy számolva, hogy a logaritmusát veszi a korpusz összes dokumentumának (*N*) és a kifejezést tartalmazó dokumentumok (df_t) hányadának (Grootendorst, 2022, p. 3.):

$$W_{t,d} = tf_{t,d} \cdot log\left(\frac{N}{df_t}\right)$$

Ezt a módszert alakítja át Grootendorst osztályalapúvá úgy, hogy a szavak fontosságát a dokumentumok klasztereire vonatkoztassa az egyes dokumentumok helyett. Az egyes klaszterekbe tartozó dokumentumokat összefűzzük, és ezt nevezzük *c* osztálynak. A kifejezésgyakoriságot ($tf_{t,c}$) itt az osztályon belül nézzük. Az inverz dokumentumgyakoriság helyett inverz osztálygyakoriságot nézünk, hogy megtudjuk, mennyi információt szolgáltat a *t* kifejezést az osztályhoz tartozó dokumentumokban. Ez úgy számolódik, hogy az adott osztályhoz tartozó átlagos szószámot (*A*) elosztjuk a *t* kifejezés korpuszban lévő gyakoriságával. Mielőtt ennek a hányadnak a logaritmusát vennénk, hozzáadunk 1-et, hogy a kimenete mindenképpen pozitív értéktartományban legyen (Grootendorst, 2022, p. 3.):

$$W_{t,c} = tf_{t,c} \cdot log\left(1 + \frac{A}{tf_t}\right)$$

Így megkapjuk a topikokhoz tartozó szóeloszlásokat minden klaszternél. Ezután a legkevésbé gyakori topik osztály alapú TDF-IDF reprezentációját összevonjuk a hozzá leginkább hasonlóval, és ezt a folyamatot iterálva lecsökkentjük a topikok számát (Grootendorst, 2022).

Az LDA dinamikus változatához hasonlóan (Blei–Lafferty, 2006) a BERTopicnak is létezik olyan változata, amely a topikok időbeli változatát képes reprezentálni. Az alapvető feltételezése a dinamikus BERTopicnak, hogy a topikok is változhatnak az idő folyamán, így vannak általánosan jellemző szavai a topikoknak, és átmenetileg jellemzőek is[5]. A dinamikus BERTopic első lépése megegyezik a statikus BERTopic modellel, majd a korábban kiszámolt globális inverz osztálygyakoriságot meghagyva, a kifejezésgyakoriságba bevonunk egy *i* időjelző paramétert (továbbiakban timestep). A dinamikus BERTopic alternatív alkalmazása, amikor nem az idő

[5] A szerző itt azt a példát említi, hogy az autókról szóló téma az „autó" és „jármű" szavakat az idő dimenziótól függetlenül tartalmazza, de míg az utóbbi években jellemző kifejezése lenne a topiknak az „önvezető" és a „Tesla", addig ezek 30 évvel korábbi autókkal kapcsolatos dokumentumokban nem jelennének meg (Grootendorst, 2022).

alapján vizsgáljuk a „helyi" reprezentációit a topikoknak, hanem a dokumentumok szerzőit, vagy a kiadó folyóiratot vonjuk be az *i* változóval (Grootendorst, 2022, p. 4.):

$$W_{t,c,i} = tf_{t,c,i} \cdot log\left(1 + \frac{A}{tf_t}\right)$$

Ahhoz, hogy figyelembe vegyük, hogy az egyes időpontokban lévő topikreprezentációk lineárisan összefüggenek, egymásra épülnek, simítást (smoothing) alkalmazhatunk a „helyi" topikreprezentációk kalkulálásánál. Először normalizáljuk a c-TF-IDF vektorokat úgy, hogy elosztjuk az L1-normával[6], így küszöbölve ki az eltérő méretű dokumentumok okozta aránytalanságot a topikok hatásában. Ezután az i. és i-1. timestepben lévő c-TF-IDF vektorok átlagát vesszük, és ezt rendeljük az i. topikreprezentációhoz, így figyelembe véve az előző timestepben lévő topikreprezentáció hatását az azt követőre. A simítás alkalmazása opcionális a modell felépítésekor (Grootendorst, 2022).

1.3.4. A modell előnyei és hátrányai

Grootendorst (2022) három különböző korpuszon hasonlította össze a BERTopic teljesítményét az LDA, NMF, CTM modellekkel és a Top2Vec két variációjával. Az eredményeket a topikkoherencia és a topikdiverzitás alapján értékelte. A topikkoherencia (coherence score) értéktartománya [-1, 1], ahol az 1 jelenti a tökéletes asszociációt a topik szavai és a topik között. A topikdiverzitás (topic diversity) [0, 1] közötti értéket vehet fel, ahol a 0 a redundáns, átfedő topikokat jelenti, az 1 pedig a változatos, elkülönülő topikokat. A BERTopic modell általánosan magas pontszámokat ért el a topikkoherenciába, míg a topikdiverzitásban bár jó értékeket ért el, a CTM leelőzte, aminek viszont kiugróan magas volt a számítási ideje az összes többi modellhez képest.

Egger és Yu (2022) tanulmányukban rövid (maximum 280 karakter hosszú) közösségi média bejegyzésekből álló korpuszon hasonlította össze az LDA, NMF, Top2Vec és BERTopic modellek teljesítményét, hangsúlyt fektetve a működésük előnyeinek és hátrányának bemutatására. Kutatásuk során a BERTopicot találták a legalkalmasabbnak az elemzésre. Az NMF-fel együtt jól

[6] Más néven Manhattan-, cityblock- vagy taxicab-távolság. Két pont távolságát (egy vektor méretét) eltérően az euklideszi távolságtól nem a pontok által kifeszített egyenes méretével határozza meg, hanem az X és Y tengely irányában megtett utak abszolút értékének összegével (Muller, 1982).

körülhatárolt témaköröket lefedő topikokat hozott létre, de a BERTopic újszerű betekintést nyújtott a korpusz tartalmára vonatkozóan a szövegbeágyazási megközelítésnek köszönhetően. A Top2Vec szintén ezt a megközelítést alkalmazza, de az általa generált topikok nagy mértékben átfedőek voltak. AZ LDA az NMF-hez hasonlóan általános témát lefedő topikokat határozott meg.

A BERTopic modell előnyeként említi Grootendorst (2022), hogy a dokumentumbeágyazás lépését szabadon választható nyelvi modellel lehet végezni, így a BERTopic performanciája folyamatosan fejlődni tud a szövegbeágyazási technikák előrehaladásával. Továbbá a dokumentumbeágyazás és a topikreprezentációk meghatározásának lépése különválasztható, így rugalmasságot biztosít a felhasználásban, megengedi, hogy eltérő előfeldolgozást használjunk a két lépésben (pl. csak a topikreprezentációk képzésénél távolítjuk el a stopszavakat, a dokumentumbeágyazásnál nem). Egger és Yu (2022) a BERTopic modell erősségei között említi a tématerületeken átívelő stabilitást és változatosságot, illetve a képességet többnyelvű korpuszok elemzésére. A szövegbeágyazás miatt nem igényel a korpusz előfeldolgozást, és képes automatikusan meghatározni a topikszámot, miközben lehetőség van a paraméter manuális megadására is.

A modell gyengeségei közé tartozik, hogy a BERTopic minden dokumentumhoz egy topikot társít, ami nem egyezik a topikmodelleknek azzal az alaptételével, hogy a dokumentumok több topik keverékeként állnak elő. Ezt valamennyire feloldja a HDBSCAN klaszterezési technika, aminek a valószínűségi mátrixa használható a topikok dokumentumbeli keveredésének meghatározására. Emellett bár a szövegbeágyazásnál figyelembe veszi a SBERT modell a szavak kontextusát is, a topikreprezentációk szózsák modellből alakulnak ki, ennek eredményeként a topikot meghatározó szavak hasonlóak, redundánsak lehetnek, így nem feltétlenül magas a hozzáadott információjuk a topikhoz (Grootendorst, 2022). További hátrány, hogy az automatikus topiszám beállítás rendszerint túl magas topikszámot generál, és nincs még beépített objektív értékelési metrikája a modellnek (Egger-Yu, 2022).

2. A KUTATÁS MÓDSZERTANA

Kutatásomban a BERTopic alkalmazását hasonlítom össze az LDA használatával. A teszteléshez Orbán Viktor angol nyelvű miniszterelnöki beszédeit használom korpuszként. Az LDA paramétereinek optimalizálása után egy BERTopic modellt hozok létre ugyanazokkal a beállításokkal, hogy minden más változatlansága mellett összehasonlítható legyen a két modell teljesítményének különbsége. Ezután A BERTopic optimalizálását végzem el, hogy bemutassam finomhangolási lehetőségeit és az ezzel elérhető teljesítménynövekedést. Az alkalmazott Python kódjaim a https://github.com/pirossara/bertopic-lda GitHub repozitóriumban találhatóak. A Függelék A. részében olvasható a használat során felmerülő technikai akadályok és azok megoldásait összefoglaló leírás, amely segítséget nyújthat későbbi elemzések, kutatások megvalósításához.

A modellek összehasonlításánál kvantitatív és kvalitatív szempontokat is figyelembe veszek. A generált topikok jóságának két legfontosabb aspektusa, hogy egy topik valóban egy összefüggő témát fed-e le (topikkoherencia), valamint, hogy az egyes topikok mennyire különbözőek vagy redundánsak (topikdiverzitás). Ezt a két aspektust mérőszámmal fogom mérni, továbbá vizsgálom a generált topikreprezentációk értelmezhetőségét, a modellek használata közben tapasztalt előnyöket és korlátokat, valamint a további lehetőségeiket.

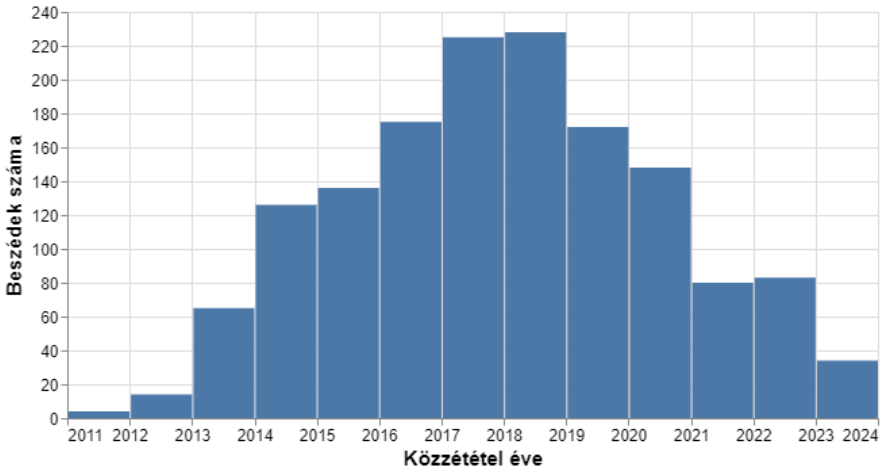
A topikkoherenciát, tehát az egy topikhoz tartozó szavak egymáshoz való hasonlóságát a coherence score mutatóval mérem. Ez a mutató a legelterjedtebb a topikmodellek jóságának vizsgálatára, hiszen azt mutatja meg, hogy a topikreprezentációk tényleg egy-egy összefüggő témát fednek-e le (Grootendorst, 2022). A coherence score-t több mutatóval is meg lehet határozni, amelyeknek eltérő számítási metódusa van. Ilyen például a UMass, a UCI és az NPMI. Előbbi a szavak dokumentumbeli együttes előfordulása alapján méri, utóbbi kettő pedig csúszóablakos metódussal vizsgálja a szópárok közötti pointwise mutual informationt, azaz PMI-t (Mifrah–Benlahmar, 2020). A PMI azt állapítja meg, hogy mennyiben tér el a szópárok együttes előfordulása a függetlenség feltételezésével várt előfordulásuktól (Bouma, 2009). Grootendorst (2022) az NPMI-t használja, ami a PMI normalizált verziójával számolt coherence score, mert az emberi ítélethez hasonló eredményt hoz a koherenciáról, viszonylag gyors számítási idővel. Így bár az optimalizálás folyamán a többi mutatót is vizsgáltam, végül a

kutatásomban én is az NPMI-t választottam coherence score-nak. A mutató a szokásokat követve a topikreprezentációk 10 legnagyobb valószínűséggel rendelkező szavára kerül kiszámításra. -1 és 1 közötti értéket vehet fel, minél nagyobb, annál jobb koherenciát mutat (Grootendorst, 2022).

A topikreprezentációk különbözőségének (továbbiakban topic diversity) mérésének is számos különböző lehetősége van. Vannak mutatók, amelyek a közös szótokeneken alapulnak, ilyen a topikpáronkénti átlagos PMI, az átlagos Jaccard-hasonlóság, a Rank-Biased Overlap és az egyedi szavak aránya. Más mutatók a valószínűségi eloszlás alapján határozzák meg a topic diversity-t, például az átlagos Log Odds Ratio és a Kullback-Leibler Divergence (Terragni et al., 2021). Grootendorst (2022) tanulmányában az egyedi szavak arányára épülő topic diversity-t használja, amely módszer rendkívül egyszerűen és hatékonyan mutatja be a topikok különbözőségét vagy redundanciáját. Dieng és társai (2020) vezették be ezt a mutatót, ami a topikreprezentációk legvalószínűbb 25 szavát összesíti és ehhez képest határozza meg az egyedi szavak arányát. Így a 0-hoz közeli értékek átfedő, az 1-hez közeli értékek különböző topikokat jelentenek.

2.1. AZ ADATOK BEMUTATÁSA

A modellek teszteléséhez használt korpusz Orbán Viktor 1483 db angol nyelvű miniszterelnöki beszédét tartalmazza, amelyeket 2011 és 2023 között tettek közzé a hivatalos



2. ábra: A beszédek száma a korpuszban éves bontásban

kormányoldalakon[7]. A 2018-ban közzétett beszédeknek a legmagasabb a száma a korpuszban (228 db, 15,4%), de ettől csak kevéssel marad el a 2017-es év (225 db, 15,2%).

A korpusznak az eredeti és az előfeldolgozott variációját is használtam az elemzés során. Az előfeldolgozás R-ben történt a *quanteda* csomag használatával. A folyamat során a központozás, speciális karakterek, szeparátorok kerültek eltávolításra, és a kötőjellel való tördelés, valamint a szavak kisbetűsre formázása történt meg.[8]

Eredeti beszédrészlet	Előfeldolgozott beszédrészlet
„Good Day, Distinguished Ladies and Gentlemen, Elections have been held in Hungary, as an outcome of which I am now an MP in the seventh parliamentary cycle. I have been in parliament for 24 years - 18 in opposition and eight in government as Prime Minister. This also justifies Churchill's saying that "politics is more dangerous than war, for in war you are only killed once"."	„good day distinguished ladies and gentlemen elections have been held in hungary as an outcome of which i am now an mp in the seventh parliamentary cycle i have been in parliament for 24 years 18 in opposition and eight in government as prime minister this also justifies churchill's saying that politics is more dangerous than war for in war you are only killed once"

2. táblázat: Példa a korpusz előfeldolgozására

A legrövidebb beszéd a korpuszban 74 szóból áll, a leghosszabb pedig 29 658 szóból, viszont 8 000-nél több szó összesen csak 14 beszédben van. A beszédek átlagosan 2 182 szó hosszúak, a leggyakoribb kategória pedig az 500-1000 szóhosszúság, a beszédek 22,3%-a (331 db) tartozik ide.

3. ábra: A beszédek eloszlása a korpuszban a hosszúságuk szerint

[7] A beszédek forrásai: https://2010-2014.kormany.hu/en, https://2015-2019.kormany.hu/en, https://miniszterelnok.hu/en/

[8] A beszédek scrape-elését és a szövegek előfeldolgozását Rakovics Zsófia, az ELTE RC2S2 kutatója végezte.

2.2. AZ LDA HASZNÁLATA PYTHONBAN

Az LDA futtatásához a népszerű *gensim* csomagot használtam Pythonban. A *gensim*[9] topikmodellek egyszerű és gyors implementálására létrehozott open-source könyvtár, amire a „topik modelling for humans" elnevezés is utal. A csomagot Radim Rehurek hozta létre, jelenleg a 4.3.0 verzió a legfrissebben elérhető belőle. Tartalmazza az LDA mellett többek között a Word2Vec, FastText és a Latent Semantic Indexing (LSI) algoritmusokat is. Az LDA elterjedtségét és népszerűségét is bizonyítja, hogy használatáról számos cikk és blogbejegyzés is született, amik lépésről lépésre végigvezetik az olvasót az implementáció folyamatán[10].

Az LDA alkalmazásához szükség van előfeldolgozásra, mert a dokumentumokat bag-of-words-ként értelmezi. Így ennél a modellnél a korpusz előfeldolgozott változatát használtam, a dokumentumokat tartalmazó listává alakítva. Az *LdaModel()* függvénynek[11] szüksége van egy *id2word* paraméterre, ami lényegében egy szótár, ami a korpusz szavaihoz azonosítót rendel. Ezt a *corpora.Dictionary(texts)* paranccsal könnyen el tudjuk érni, ha a paramétereként megadjuk a korpuszunkat. Az *LdaModel() corpus* paraméterébe a korpuszt bag-of-words formátumban kell megadni. Az átalakításhoz megfelelő módszer a *[id2word.doc2bow(text) for text in texts]*[12], amivel a dokumentumainkat tuple-öket tartalmazó listává alakítjuk át, ahol az egyes tuple-ök a szó (token) azonosítóját és dokumentumbeli számosságát tartalmazzák (ez a term document frequency). Továbbá be kell állítani a topikszámot (*num_topic*), és a megismételhetőség kedvéért a *random_state*-et is.

A generált topikreprezentációkat az *lda_model.print_topics()* és az *lda_model.show_topics()* parancsokkal lehet kiíratni. Így megkapjuk a topikokhoz tartozó szavakat és a valószínűségüket. Mindkét metódus esetén a *num_words* paraméterben határozhatjuk meg, hogy a hány legvalószínűbb topikszót szeretnénk látni, az alapbeállítása 10 szó. Fontos beállítani a *num_topics* paramétert is, ami azt befolyásolja, hogy hány topikreprezentációt írjon ki a parancs (a valós topikszámunknál többet is beállíthatunk), mert alapértelmezett beállításként a *show_topics()* csak 10, véletlen választott topikot ad vissza.

[9] Hivatalos honlapja: https://radimrehurek.com/gensim/ (megnyitva: 2024.04.06.)
[10] Például: https://bennett-holiday.medium.com/a-step-by-step-guide-to-writing-an-lda-program-in-python-690aa99119ea (megnyitva: 2024.03.28.)
[11] https://radimrehurek.com/gensim/models/ldamodel.html (megnyitva: 2024.03.28.)
[12] https://tedboy.github.io/nlps/generated/generated/gensim.corpora.Dictionary.doc2bow.html (megnyitva: 2024.03.28.)

2.2.1. Az optimalizált LDA modell bemutatása

A korpusz előfeldolgozása során csak formázás történt, a stopszavak eltávolítását az elemzés közben végeztem. Így jól látszik, hogy az LDA első eredménye redundáns topikokat eredményezett, amelyek névmásokból, segédigékből és a korpuszra globálisan jellemző szavakból álltak össze (*3. táblázat*). Ez rámutatott arra, hogy az általános stopszavakon túl a dokumentum leggyakoribb szavait is el kell távolítani annak érdekében, hogy egymástól különböző és koherens topikokat kapjunk. A szakirodalomban vitatott téma az általános stopszavak eltávolításának kérdése (ide értendőek a névmások, segédigék, kötőszavak), míg többen rámutatnak a fontosságára a modell teljesítményének, hatékonyságának és pontosságának javítása érdekében (Silva-Ribeiro, 2003; Kaur-Buttar, 2018), addig mások vitatják az érdemi hatását, főként az idő- és munkaigényességének tekintetében, és csak a korpusz-specifikus stopszavak, tehát az akár önálló jelentéssel bíró, de túl gyakran előforduló szavak eltávolítását ajánlják (Schofield et al., 2017). Én az általános stopszavak és az azokon kívüli leggyakoribb szavak eltávolítása mellett döntöttem. Az általános stopszavakat egyszerű volt eltávolítanom az *nltk* csomag *stopwords* funkciójával, a további leggyakoribb szavak számát pedig a coherence score-ra tekintettel határoztam meg.

1. topik		2. topik		3. topik		4. topik		5. topik	
also	0,013	hungary	0,011	hungarian	0,012	hungarian	0,011	hungary	0,012
hungary	0,011	also	0,010	hungary	0,010	us	0,008	people	0,011
us	0,010	people	0,009	europe	0,009	hungary	0,008	hungarian	0,010
one	0,008	hungarian	0,008	one	0,008	also	0,008	also	0,009
hungarian	0,008	one	0,008	also	0,008	people	0,007	one	0,008
people	0,007	would	0,006	people	0,008	europe	0,006	us	0,008
european	0,006	us	0,006	us	0,008	must	0,006	european	0,007
europe	0,006	european	0,005	european	0,007	european	0,005	would	0,006
would	0,005	must	0,005	must	0,006	one	0,005	eurpe	0,006
must	0,005	like	0,005	like	0,006	would	0,005	must	0,006
6. topik		7. topik		8. topik		9. topik		10. topik	
hungarian	0,010	hungarian	0,012	hungary	0,013	us	0,010	hungary	0,016
hungary	0,009	hungary	0,009	also	0,011	hungarian	0,009	also	0,010
also	0,009	people	0,008	hungarian	0,008	hungary	0,009	hungarian	0,009
one	0,009	european	0,007	european	0,008	people	0,008	european	0,008
us	0,008	would	0,007	one	0,008	european	0,008	people	0,008
european	0,008	us	0,007	europe	0,007	one	0,008	europe	0,007
would	0,007	one	0,007	would	0,007	must	0,007	one	0,007
people	0,006	also	0,007	us	0,007	also	0,007	would	0,006
government	0,005	must	0,006	like	0,005	would	0,006	country	0,005
europe	0,005	like	0,006	people	0,005	europe	0,005	important	0,005

3. táblázat: LDA topikreprezentációk stopszavak eltávolítása nélkül

Az aposztróffal való tördelést a stopszavakat meghatározó függvényen belül végeztem, amikor a beszédeket szótokenekre vágtam. Két metódust teszteltem le, a *gensim* csomag *simple_preprocess()* funkcióját, és a regular expression segítségével meghatározott tördelést, a *re.split(" |'", doc)* parancsot. Mindkét módszer jobban teljesített, mintha nem végeztem volna tördelést az aposztrófok mentén, ugyanis az *nltk* stopszó listája nem tartalmaz aposztrófos alakokat. A *simple_preprocess()* használatával jobb modelleredményeket értem el, így azt használtam.

Az LDA optimalizálásának legfontosabb aspektusa alapvetően a topikszám meghatározása. Az optimális topikszámot ugyanakkor befolyásolja a stopszó lista hosszúsága (hiszen ezáltal változik a korpusz), így az algoritmusomban 50, 75, 95, 100, 150 és 200 leggyakoribb szó eltávolítása esetén is vizsgáltam az 1-20 topikszámmal beállított LDA modell teljesítményét a coherence score-ok szempontjából. Az optimalizálást még az NPMI coherence score mutató kiválasztása előtt végeztem, így 4 topikkoherencia mutató („c_v", „c_npmi", „c_uci" és „c_mass") eredményét együttesen vizsgáltam, és bár az abszolút értékük különbözött, a modellek közötti relatív eltérések összhangban voltak. A generált modellek közül a két legjobban teljesítőt mutatom be (*4. táblázat*).

Modell	Topikszám	Eltávolított gyakori szavak száma	Coherence score	Topic diversity
„A"	7	50	-0.0057	0.2857
„B"	9	95	-0.0147	0.3822

4. táblázat: „A" és „B" LDA modell jellemzői

Mivel a gyakori szavak eltávolításával körültekintően kell eljárni az esetleges információveszteség elkerülése érdekében, mindenképp fontosnak tartottam kvalitatív szempontot is bevonni a modellválasztásba, tehát, hogy emberi szemmel melyik modellnek értelmezhetőbbek a topikreprezentációi. Bár az „A" modell jobb eredményt hozott a coherence score tekintetében, a generált topikreprezentációk nem fedtek le jól körülhatárolható témaköröket, a topikszavak általánosak voltak és nem összefüggőek (*5. táblázat*). A „B" modellben már több érdekes, nagyobb jelentéssel bíró és kevésbé általános kifejezés jött elő a topikszavak között (*6. táblázat*), így ezt a modellt választottam.

1. topik		2. topik		3. topik		4. topik	
year	0.0034	central	0.0033	great	0.0033	make	0.0030
always	0.0027	come	0.0029	everyone	0.0030	way	0.0029
believe	0.0027	believe	0.0027	year	0.0028	great	0.0028
way	0.0025	may	0.0027	know	0.0028	believe	0.0027
life	0.0025	right	0.0026	something	0.0026	come	0.0027
know	0.0025	life	0.0026	right	0.0026	may	0.0027
president	0.0025	things	0.0026	always	0.0025	politics	0.0026
something	0.0024	nation	0.0025	therefore	0.0025	therefore	0.0026
may	0.0024	cooperation	0.0024	thank	0.0024	life	0.0026
come	0.0024	thank	0.0024	things	0.0023	issue	0.0025

5. topik		6. topik		7. topik	
know	0.0032	year	0.0037	thank	0.0031
question	0.0029	thank	0.0031	may	0.0029
make	0.0028	take	0.0027	life	0.0027
year	0.0027	therefore	0.0027	therefore	0.0027
great	0.0027	central	0.0026	take	0.0027
take	0.0027	per	0.0026	nation	0.0027
way	0.0027	system	0.0025	always	0.0026
let	0.0026	cent	0.0024	come	0.0025
everyone	0.0026	something	0.0023	right	0.0024
right	0.0025	everyone	0.0023	know	0.0024

5. táblázat: „A" LDA modell topikreprezentációi

1. topik		2. topik		3. topik		4. topik		5. topik	
let	0.0027	christian	0.0025	migration	0.0025	let	0.0026	serbia	0.0025
support	0.0025	still	0.0022	different	0.0022	made	0.0023	party	0.0022
eu	0.0023	live	0.0021	christian	0.0020	support	0.0023	children	0.0021
thing	0.0022	however	0.0021	history	0.0019	election	0.0022	budapest	0.0021
money	0.0022	states	0.0020	still	0.0019	migration	0.0021	thing	0.0021
children	0.0021	means	0.0020	times	0.0019	agreement	0.0021	difficult	0.0020
far	0.0020	success	0.0020	family	0.0019	still	0.0020	let	0.0020
still	0.0020	party	0.0019	let	0.0019	look	0.0020	family	0.0020
already	0.0019	lives	0.0018	crisis	0.0019	already	0.0019	migration	0.0019
successful	0.0019	brussels	0.0018	far	0.0018	culture	0.0019	support	0.0019

6. topik		7. topik		8. topik		9. topik	
brussels	0.0022	let	0.0024	let	0.0032	look	0.0026
let	0.0022	states	0.0023	energy	0.0022	already	0.0022
history	0.0020	long	0.0022	successful	0.0022	respect	0.0021
honourable	0.0020	look	0.0022	look	0.0022	western	0.0020
order	0.0019	end	0.0021	order	0.0021	continue	0.0020
still	0.0019	number	0.0021	member	0.0021	migration	0.0019
clear	0.0019	made	0.0021	migration	0.0021	development	0.0019
live	0.0018	growth	0.0021	far	0.0021	issues	0.0019
talk	0.0018	decision	0.0021	development	0.0020	view	0.0018
foreign	0.0018	brussels	0.0021	states	0.0019	money	0.0018

6. táblázat: „B" LDA modell topikreprezentációi

2.3. A BERTOPIC HASZNÁLATA PYTHONBAN

A BERTopic alkalmazásához a Maarten Grootendorst által fejlesztett *bertopic*[13] csomagot lehet használni Pythonban. A jelenlegi legfrissebben elérhető verziója a 0.16.0. A legelső verzióját 2020-ban, a modellt bemutató tanulmány megjelenése előtt 2 évvel fejlesztette ki Grootendorst.

A BERTopic alkalmazása rendkívül rugalmas. A modell korábban ismertetett lépései (szövegbeágyazás, dimenziócsökkentés, klaszterezés és topikreprezentációk létrehozása) tetszőleges módszerrel elvégezhetőek a megfelelő beállításokkal. Az alapértelmezett technikák (pl. a UMAP a dimenziócsökkentéshez, a HDBSCAN a klaszterezéshez) a felhasználó igénye szerint módosíthatóak az úgynevezett pipeline paraméterek beállításával. Ez a rugalmasság a modell egyik legfontosabb előnye, így tud lépést tartani az NLP fejlődésével, valamint alkalmazkodni a felhasználó igényeihez és a korpuszának sajátosságaihoz.

A szövegbeágyazáshoz elsősorban az SBERT sentence-transformereket[14] ajánlják. Az all-MiniLM-L6-v2 az alapértelmezetten használt módszer közülük, ugyanis ez különösen jól teljesít a dokumentumok szemantikai hasonlóságainak meghatározásában. Ugyanakkor a BERTopic hivatalos oldalán bemutatják további szövegbeágyazási technikák használatát is, például a Scikit-Learn Embeddings, a Spacy, a Gensim és az OpenAI lehetőségeit. A technikát a *BERTopic()* függvény *embedding_model* paraméterében lehet állítani. [15]

A dimenziócsökkentéshez alapértelmezetten az UMAP-ot használják, ami a lokális és globális vonásokat is kiemelkedően jól megőrzi az alacsonyabb dimenziószámban is, de ez szabadon kicserélhető például a széles körben használt PCA-ra, ami gyorsabb futást eredményezhet. A PCA és további dimenziócsökkentő eljárások is megtalálhatóak a *sklearn.decomposition* csomagban. A választott eljárást a *BERTopic()* függvény *umap_model* paraméterében lehet állítani, bármilyen módszert választunk. A dimenziócsökkentés lépése akár teljesen el is hagyható, ha például kis méretű a korpuszunk és a dimenziócsökkentésnek nincsen jelentős

[13] Hivatalos repozitóriuma a https://github.com/MaartenGr/BERTopic (megnyitva: 2024.04.06.), a részletes ismertetőjét, dokumentációját és a szerző ajánlásait pedig itt lehet elolvasni: https://maartengr.github.io/BERTopic/index.html (megnyitva: 2024.04.06.)
[14] https://www.sbert.net/docs/pretrained_models.html (megnyitva: 2024.03.27.)
[15] https://maartengr.github.io/BERTopic/getting_started/embeddings/embeddings.html (megnyitva 2024.03.27.)

hatása a klaszterezés minőségére. Ezt úgy tudjuk elérni, ha a *BaseDimensionalityReduction()*-t adjuk meg a *umap_model* paraméterben.[16]

A klaszterezés rendkívül fontos lépése a modellnek, hiszen minél jobb klasztereket hozunk létre, annál pontosabbak lesznek a topikreprezentációink. Az alapértelmezett technika ehhez a lépésben a HDBSCAN, mert jól teljesít a különböző sűrűségű klaszterek megtalálásában. Ugyanakkor nem biztos, hogy a felhasználó korpuszán minden esetben ez a módszer biztosítja a legjobb teljesítményt, így könnyen kicserélhető az elterjedt k-means technikára például. A *BERTopic() hdbscan_model* paraméterében tudjuk ezt meghatározni.[17]

A HDBSCAN használatával (de ilyen a DBSCAN és az OPTICS is például) egy outlier klaszter, és ezáltal egy „-1" jelölésű outlier topik és létrejön, amely azokat a dokumentumokat foglalja magába, amelyek egyik topikba sem illenek. Ezáltal a létrejövő topikok összefüggőbbek és homogénebbek, hiszen a modell nem kényszerít bele minden dokumentumot egy-egy topikba. Lehetőség van az outlier dokumentumok számának egyszerű csökkentésére is a *topic_model.reduce_outliers(docs, topics)* [18] paranccsal. A módszer finomhangolható további paraméterek megadásával.

2.3.1. Optimalizálási lehetőségek

A teljesítmény optimalizálása és a futási idő lecsökkentése érdekében a szövegbeágyazást el lehet végezni előre, így nem kell újragenerálja a modell a beágyazásokat minden illesztésnél, ami önmagában is időigényes folyamat. A választott *embedding_model* megadása után az *embeddings = embedding_model.encode(docs, show_progress_bar=True)* kóddal lehet ezt megtenni. A dokumentumbeágyazás futási ideje hosszú, de a *show_progress_bar* bekapcsolásával nyomon tudjuk követni a folyamatot.[19]

A szövegbeágyazáshoz a korábbiakban említett sentence transformereket használja a BERTopic, tehát a szavakat nem külön helyezi el a vektortérben, hanem mondatok vagy egész

[16] https://maartengr.github.io/BERTopic/getting_started/dim_reduction/dim_reduction.html (megnyitva: 2024.03.27.)
[17] https://maartengr.github.io/BERTopic/getting_started/clustering/clustering.html (megnyitva: 2024.03.27.)
[18] https://maartengr.github.io/BERTopic/getting_started/outlier_reduction/outlier_reduction.html (megnyitva: 2024.04.06.)
[19] https://maartengr.github.io/BERTopic/getting_started/tips_and_tricks/tips_and_tricks.html#pre-compute-embeddings (megnyitva: 2024.03.30.)

bekezdések reprezentációit hozza létre. Amikor egész dokumentumokat adunk meg bemeneti értékként az átalakításhoz, ezeknek egy lerövidített részét használja csak a modell. Így a szerző ajánlja a dokumentumok mondatokra vágását a beágyazások elkészítése előtt.[20] Ellenben nekem a coherence score és topic diversity szempontjából rosszabb eredményt hozott, amikor a mondatok beágyazását készítettem el a dokumentumok helyett, így érdemes letesztelni mindkét verziót a modellek optimalizálása során, és a korpuszunkon jobban teljesítő opciót választani.

A reprodukálhatóság kedvéért szokás szerint *random_state*-et tudunk beállítani. Ez a paraméter a *umap_model*-en belül adható meg (pl*. umap_model = UMAP(random state=42)*), ugyanis ez az a lépés, ami minden futtatásnál eltérő eredményt adhat.[21]

Az általános és korpusz-specifikus stopszavak eltávolítására, illetve csökkentésére három különböző módszer is van a BERTopic-ban: további pipeline paraméterként adhatunk meg *vectorizer_model*-t, *ctfidf_model*-t és *representation_model*-t is, amik a topikreprezentációk minőségének javírására, az interpretálhatóságuk fejlesztésére szolgálnak.[22]

A *vectorizer_model*-nek beállíthatjuk a *CountVectorizer()*-t (a *sklearn.feature_extraction.text* csomagból), aminek a stop_words paraméterében az *"english"* megadásával az alapértelmezett angol nyelvű stopszavakat szűrhetjük ki, vagy bármilyen listát megadhatunk itt a kiszűrni kívánt szavakról. További paramétere a max_df és min_df, amikkel a document frequency alsó és felső határát állíthatjuk be, tehát eldobja azokat a szavakat, amik a dokumentumoknak ezeknél kisebb vagy nagyobb arányában fordulnak elő.

Az osztály alapú TF-IDF, aminek a hátterét és működését az 1.3. fejezetben mutattam be, figyelembe veszi, hogy miben különböznek az egyes klaszterekben lévő dokumentumok a többi klaszterben lévőtől, így segítve a topikreprezentációk létrejöttét. Alapvetően a BERTopic szózsák modellként határozza meg a topikreprezentációkat és a c-TF-IDF-fel rendel a szavakhoz súlyokat, így a topikok a modell tanítás (model training) után is frissíthetőek az re-training szükségessége nélkül. Külön beállítás nélkül is ez hasznosul a *BERTopic()*-ban, viszont a

[20] https://maartengr.github.io/BERTopic/getting_started/tips_and_tricks/tips_and_tricks.html#document-length (megnyitva: 2024.03.30.)
[21] https://maartengr.github.io/BERTopic/getting_started/best_practices/best_practices.html#preventing-stochastic-behavior (megnyitva: 2024.03.30.)
[22] https://maartengr.github.io/BERTopic/getting_started/tips_and_tricks/tips_and_tricks.html#removing-stop-words (megnyitva: 2024.03.30.)

finomhangolásánál lehetőség van a korpusz-specifikus szavak előfordulásának csökkentésére. Tehát a *ctfidf_model*-nek egyszerűen beállíthatjuk a *ClassTfidfTransformer(reduce_frequent_words = True)*-t.

A harmadik lehetőség a *representation_model* parameternél a *KeyBERTInspired()* megadása. A modell egyik újítása (v0.14 óta elérhető), hogy representation model-t is be lehet állítani benne, amivel az alapvetően létrehozott topikreprezentációk tovább finomíthatóak. A KeyBERTInsired() a kulcsszavak gyors kiválogatásában segít, így előnyös az általános stopszavak eltávolítására. Amennyiben állítunk be *representation_model*-t, az *embedding_model*-t is meg kell adni a *BERTopic()*-ban, akkor is, ha előzetesen elkészítettük a szövegbeágyazásokat. Ez a beállítás segíthet a reprezentációk javításában, de meghosszabbítja a futási időt.

A *BERTopic()*-on belül, a pipeline paramétereken túl további paraméterek állíthatunk be a modell finomhangolására.[23] A legfontosabb paraméter az *nr_topics*, amivel a topikok számát adhatjuk meg, mint az LDA-ban, viszont a BERTopicban lehetőség van automatikus topikszámgenerálás beállítására is az *'auto'* bemenettel. Fontos figyelembe venni, hogy a *nr_topics* állításával utólag vonja össze vagy választja szét a topikokat a modell, hogy elérje a felhasználó által beállított topikszámot. A *min_topic_size*-zal tudjuk meghatározni, hogy mekkora legyen a topikok minimális mérete, tehát hány dokumentumból álljanak elő. A paraméter alapbeállítása 10, de nagy korpusz esetében ezt érdemes magasabbra állítani, figyelve arra, hogy ha túl magasra állítjuk a számot, lehet, hogy akár egy topikot sem képes előállítani a modell. A *top_n_words* paraméterrel tudjuk beállítani, hogy a hány legvalószínűbb topikszót jelenítse meg topikonként. Ezt a paramétert érdemes 30 alatt tartani, mert efölött a reprezentációk nagy mértékben veszítenek a koherenciájukból. A coherence score-t alapértelmezetten a legvalószínűbb 10 topikszóra szokás meghatározni (Röder et al., 2015). Az *n_gram_range* a minimum és maximum n-gramokat állíthatjuk be, így például az (1, 2) beállítással akár kétszavas kifejezéseket is kaphatunk a topikreprezentációkban. A *calculate_probabilities* paraméter *True*-ra állításával megkaphatjuk a topikok valószínűségét az egyes dokumentumokban, viszont érdemes figyelembe venni, hogy ennek nagy a számítási igénye.

[23] https://maartengr.github.io/BERTopic/getting_started/parameter%20tuning/parametertuning.html#bertopic (megnyitva: 2024.03.30.)

2.3.2. Az LDA-beállítású BERTopic modell bemutatása

Az első BERTopic modell beállításainál arra törekedtem, hogy a lehető legjobban közelítsem az LDA-ban optimálisnak talált beállításokat. Ennek három fő aspektusa volt: ugyanolyan előfeldolgozással készített korpuszt használni, ugyanazokat a stopszavakat eltávolítani, és ugyanazt a topikszámot beállítani.

Bár a BERTopic nem igényel előfeldolgozást, lehetőség van az alkalmazására, hiszen csak a szövegbeágyazások készítésénél fontos, hogy az eredeti szövegeket adjuk meg. Így a modell illesztésénél meg tudtam adni az előfeldolgozott korpuszt, ami a coherence score-ra is pozitív hatással volt.

A korpusz-specifikus stopszavak kiszűréséhez a *CountVectorizer()*-t használtam, a *stop_words* paraméterében ugyanazt a listát adtam meg, amit az LDA modellnél is. Tehát kiszűrtem az *nltk* csomagban alapértelmezetten beállított angol nyelvű stopszavakat, az ezek után maradó leggyakoribb 95 szót és a honlapok neveit (azokat a tokeneket, amelyik tartalmazták a „.hu” kifejezést).

Az azonos topikszámot a *BERTopic() nr_topics* paraméterének 10-re állításával értem el, mert ebbe beleszámolódik az outliereket tartalmazó topik is. Fontos kiemelni, hogy a felhasználó által beállított topikszámot a generált topikok összevonásával éri el a modell. A topikszámot közvetetten a *HDBSCAN() min_cluster_size* paraméterének finomhangolásával is lehet állítani.

A szövegbeágyazásnál a dokumentumokat adtam meg egységeknek, és nem a mondatokat az ajánlás ellenére, mert coherence score szempontjából jobban teljesített az előbbi. A megismételhetőség és ellenőrizhetőség kedvéért a *random_state* paramétert 42-re állítottam a *UMAP()*-ban.

2.3.3. Az optimalizált BERTopic modell bemutatása

A másik BERTopic modellnél az volt a célom, hogy felfedezzem a további optimalizálási lehetőségeket és megtaláljam a BERTopic modellek közül a legjobban teljesítőt. A legfőbb eltérése az LDA-beállítású modelltől, hogy a BERTopic saját, beépített módszereit használtam a stopszavazáshoz, és a topikszám megtaláláshoz az automatikus beállítást használtam a HDBSCAN finomhangolásával.

A korpusz-specifikus stopszavak eltávolítására a manuálisan megadott lista helyett a korábban ismertetett, BERTopic-nál elérhető metódusokat használtam: *CountVectorizer(stop_words = 'English')* és *ClassTfidfTransformer(reduce_frequent_words = True)*. A topikreprezentációk javítása és a stopszavak előfordulásának csökkentésére a KeyBERTInspired() beállítása is ajánlott a *representation_model* paraméterben, de ezzel a beállítással átfedőbb topikok jöttek létre, amelyekben sokszor szerepeltek a korpusz gyakori szavai (pl. „hungary", „hungarians"), míg az előbbi két módszer együttes használatával csak egy olyan topik jött létre, amely ezeket a szavakat foglalta magába. A túl gyakori szavak kiszűrésére a *CountVectorizer() min_df* és *max_df* paraméterének hangolását is teszteltem, amikkel azt lehet megadni, hogy milyen dokumentumbéli előfordulási arány (document frequency) alatt és fölött ignorálja a szótokeneket. Ezekkel nem értem el olyan jó topikreprezentációkat, mint a *CountVectorizer(stop_words = 'English')* és *ClassTfidfTransformer(reduce_frequent_words = True)* kombinációjával.

A választott beállításokkal előkerültek a topikreprezentációkban olyan szótokenek, amelyek egyértelműen stopszavak voltak, például aposztróf nélküli segédigék („weve", „theres", „theyre", „dont", „isnt"), további önálló jelentéssel nem bíró segédigék, határozószók („also, „well", „like", „would", „most"), scrape-elés során bekerült metaadatok („sz00f6vegtestchar", „sz00f6vegtest", „span"), és monogramok az interjú felépítésű dokumentumokból, amiket a párbeszéd jelölésére használtak („vo" – Orbán Viktor, „pcs" – Csermely Péter, „gik" – Kiss Gábor István). Ezeket az alapértelmezett angol stopszólistával (*stopwords.words('english')*) összefűzve a CountVectorizer() stop_words paraméterébe megadva eltávolítottam, így Schofield és társai (2017) ajánlását követtem, akik tanulmányukban az eredmények alapján meghatározott korpusz-specifikus stopszavak eltávolítását javasolták.

A topikszám automatikus megtalálásához a *BERTopic() nr_topics* paraméterét *'auto'*-ra állítottam. Az így megtalált topikszám minden esetben 34+1 topikot generáltak, bármilyen beállítással. Nem volt rá hatással a stopszavazás a *ClassTfidfTransformer(reduce_frequent_words = True)* és a *CountVectorizer(stop_words='english')* használatával sem, ami azt mutatja, hogy ezek a metódusok csak a topikreprezentációk megalkotásánál szűrik ki a szavakat, nem a topikok generálásánál. A magas topikszámot Egger és Yu (2022) a BERTopic hátrányaként említi tanulmányában, amelyet a szózsák modell helyetti szövegbeágyazást alkalmazó megközelítésnek tulajdonítanak.

A BERTopic topikszámának finomhangolását a *HDBSCAN() min_cluster_size* paraméterének állításával lehet elérni, amelynek optimális mértékét az elemzett korpusz befolyásolja. A paramétert 15-re állítva 4 topikot adott vissza a modell, amik között nem volt a szokásos -1 jelölésű outlier topik. A modell ezzel sokkal jobb coherence score-t és topic diversity-t ért el, mint korábban, de kevésnek ítéltem a 4 topikot ezért csökkentettem a *min_cluster_size* paraméteren, hogy megengedjem a modellnek, hogy több topikot generáljon. A min_cluster_size-ot 10-re csökkentve 16+1 topik jött létre, és bár a coherence score javult, a topic diversity csökkent.

Az UMAP modellben beállítottam a random_state-et szintén 42-re, a megismételhetőség érdekében. Az *n_neighbors* paraméterét 5-re, az *n_components*-et pedig 15-re állítottam. Ezekkel azt lehet megadni, hogy mekkora legyen a közelítéshez használt szomszédos minták száma, és a dimenziócsökkentés utáni dimenziószáma a beágyazásoknak. Ezek a paraméterek is hatással vannak a topikszámra. A többdimenziós távolság mérésére az alapértelmezett koszinusz metrikát használtam.

A szövegbeágyazáshoz ennél a modellnél is az all-MiniLM-L6-v2 sentence transformert használtam az SBERT transzformerek közül, és nem alkalmaztam mondat tördelést, mert anélkül jobb eredményeket generált a modell. Az *n_gram_range* paramétert a *BERTopic()*-ban (1, 2)-re állítottam, tehát megengedtem, hogy két szóból álló szótokenek is bekerüljenek a topikreprezentációkba, de ez végül nem fordult elő. A *top_n_words*-t 10-re állítottam a coherence score kiszámításához, és 25-re a topic diversity megállapításához.

3. EREDMÉNYEK

A korábban bemutatott három topikmodell topikreprezentációinak részletes bemutatását és eredményeik összevetését tartalmazza ez a fejezet. A bemutatott vizualizációk kapcsán kitérek az LDA és BERTopic Python implementációjának topikvizualizációs lehetőségeire.

A topikokra való hivatkozás megkönnyítésének és megfoghatóbbá, értelmezhetőbbé tételének érdekében felcímkéztem őket. Törekedtem a szavak által lefedett téma megfelelő meghatározására, de mivel a dolgozatom célja nem a beszédek tartalmi elemzése, hanem a modellek teljesítményének mérése és a generált topikok hasonlóságának összevetése, így nem tértem ki kutatásomban topikelnevezések optimalizálására. A megbízható tartalmi következtetésekhez további kutatás szükséges.

3.1. AZ OPTIMALIZÁLT LDA MODELL EREDMÉNYEI

Az optimalizált LDA 9 topikreprezentációja a *7. táblázatban* látható, topikonként megtalálhatóak benne a topikokhoz tartozó legvalószínűbb szavak és a topikvalószínűségük. Ennek a modellnek a topikneveinél a legfontosabb kiemelni, hogy nem fedik le megfelelően a topikok tartalmát, csak támpontként értelmezendőek. A redundanciából és alacsony szintű topikkoherenciából adódó felcímkézési nehézség megoldására a topikok jellemzőbb szavainak és az egyedi, más topikban elő nem forduló szavainak kombinációjaként adtam meg az elnevezéseket. A manuális címkézés nehézsége is azt mutatja, hogy a modell nem tudott olyan topikokat generálni, amik emberi szemmel is egy-egy jól körülhatárolt, összefüggő témakört fednek le.

Az LDA modellek vizualizációs eszközét a *pyLDAvis*[24] csomag biztosítja. A Carson Sievert és Kenny Shirley által fejlesztett R csomag[25] Python portját Ben Mabey készítette, jelenleg a 3.4.1 verziója érhető el. A *prepare()* paranccsal egyszerűen elkészíthető a két részből álló modellvizualizáció. Paraméterként a modellt, a korpuszt (bag-of-words formátumban) és a szótárat kell megadni, amelyek specifikációjáról a 2.2. alfejezetben írtam.

[24] https://pyldavis.readthedocs.io/en/latest/modules/API.html# (megnyitva: 2024.04.03.)
[25] https://cran.r-project.org/web/packages/LDAvis/LDAvis.pdf (megnyitva: 2024.04.03.)

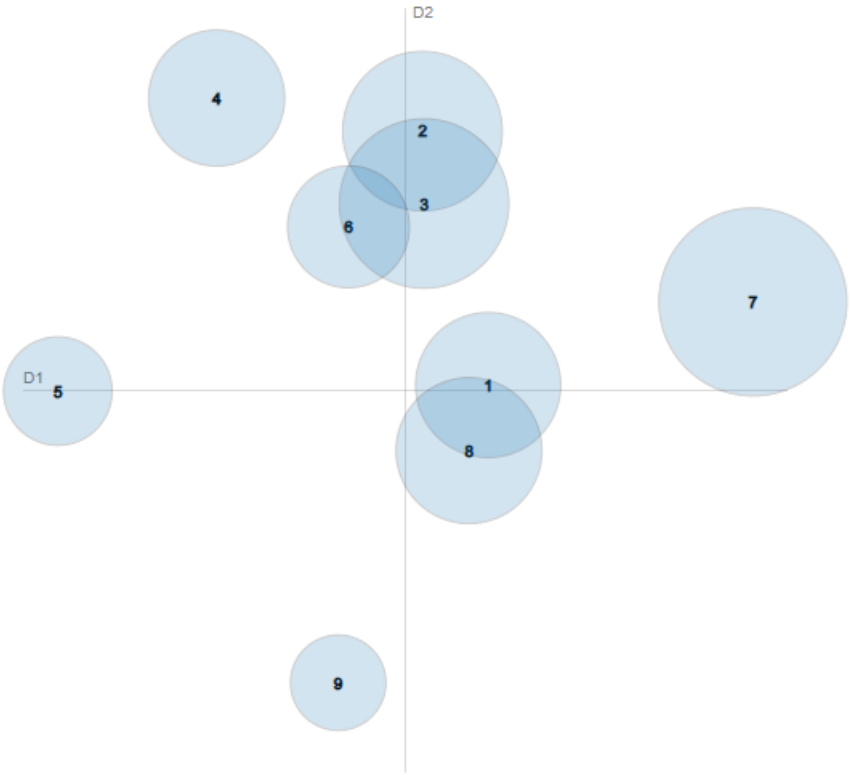
1: Támogatás & EU		2: Kereszténység & siker		3: Migráció & krízis		4: Támogatás & választás		5: Szerbia & nehézség	
let	0.0027	christian	0.0025	migration	0.0025	let	0.0026	serbia	0.0025
support	0.0025	still	0.0022	different	0.0022	made	0.0023	party	0.0022
eu	0.0023	live	0.0021	christian	0.0020	support	0.0023	children	0.0021
thing	0.0022	however	0.0021	history	0.0019	election	0.0022	budapest	0.0021
money	0.0022	states	0.0020	still	0.0019	migration	0.0021	thing	0.0021
children	0.0021	means	0.0020	times	0.0019	agreement	0.0021	difficult	0.0020
far	0.0020	success	0.0020	family	0.0019	still	0.0020	let	0.0020
still	0.0020	party	0.0019	let	0.0019	look	0.0020	family	0.0020
already	0.0019	lives	0.0018	crisis	0.0019	already	0.0019	migration	0.0019
successful	0.0019	brussels	0.0018	far	0.0018	culture	0.0019	support	0.0019

6: Brüsszel & külföld		7: Hosszútávú & növekedés		8: Energia & fejlődés		9: Nyugat & problémák	
brussels	0.0022	let	0.0024	let	0.0032	look	0.0026
let	0.0022	states	0.0023	energy	0.0022	already	0.0022
history	0.0020	long	0.0022	successful	0.0022	respect	0.0021
honourable	0.0020	look	0.0022	look	0.0022	western	0.0020
order	0.0019	end	0.0021	order	0.0021	continue	0.0020
still	0.0019	number	0.0021	member	0.0021	migration	0.0019
clear	0.0019	made	0.0021	migration	0.0021	development	0.0019
live	0.0018	growth	0.0021	far	0.0021	issues	0.0019
talk	0.0018	decision	0.0021	development	0.0020	view	0.0018
foreign	0.0018	brussels	0.0021	states	0.0019	money	0.0018

7. táblázat: Optimalizált LDA modell topikreprezentációi

A függvény alapvetően az előfordulási gyakoriságuk sorrendjében számozza a topikokat, a *sort_topics* paramétert *False*-ra állítva kaphatjuk meg a topikok eredeti sorrendjét. Az *enable_notebook()* paranccsal engedélyezhető a vizualizáció D3 megjelenítése IPython notebookban. A vizualizáció tartalmaz egy topikközi távolságtérképet, amely a topikokat körökként ábrázolja, középpontjukat a topikok távolsága adja meg, amelyet többdimenziós skálázással két dimenzióra vetít, a körök méretét pedig a topikok korpuszbéli előfordulási gyakorisága adja meg. Ezen túl sávdiagramot is készít az egyes topikokra jellemző szavakról és azok topikbéli, valamint korpuszbéli valószínűségéről. Ez utóbbi vizualizáció interaktív, manuálisan választható, hogy melyik topik szavait szeretnénk megtekinteni, így a dolgozatomban ezt nem mellékelem, de a GitHub repozitóriumba feltöltött jupyter notebookot lefuttatva megtekinthető. A megjelenített topikszavak számát az *R* paraméterben tudjuk megadni (Sievert – Shirley, 2014). A *pyLDAvis.prepare()* paranccsal generált topikközi távolságtérképen (*4. ábra*) az látható, hogy az 1. és 8. topik („Támogatás & EU" és „Energia & fejlődés" elnevezésűek) valamint a 2., 3. és 6. topik („Kereszténység & siker", „Migráció & krízis", „Brüsszel & külföld" elnevezésűek) vannak a legközelebb egymáshoz. A topikok

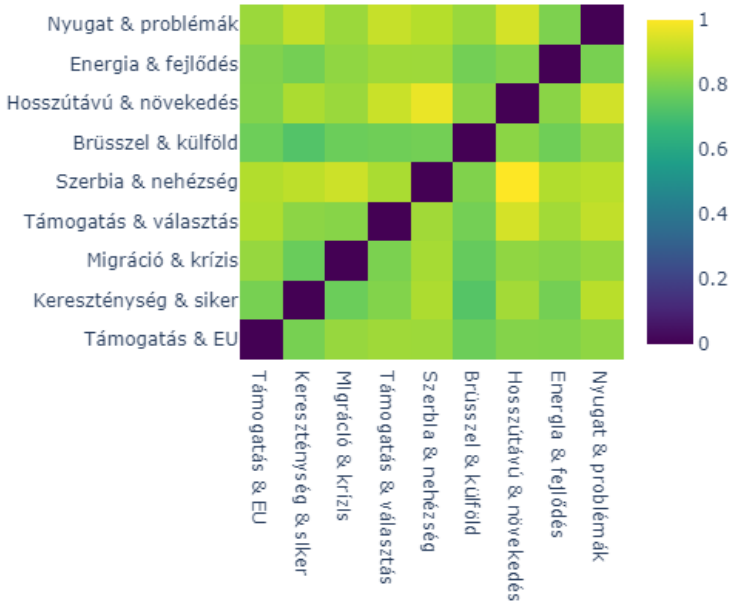
arányában nincsen kiugró eltérés, a 7. topik („Hosszútávú & növekedés”) a legnagyobb, tehát a korpuszban legtöbbet előforduló, míg a 9. („Nyugat & problémák”) a legritkább topik.



4. ábra: Optimalizált LDA modell topikközi távolságtérképe

A *gensim*ben nincsen beépített topikvizualizációs parancs, de a honlapjukon biztosítanak egy kódrészletet, amivel generálható hőtérkép a topikok hasonlóságának bemutatására[26]. Az algoritmus a *diff()*[27] parancsra épül, amely alapvetően két modell összevetésére szolgál, de amennyiben másik modell helyett önmagát adjuk meg (*lda_model.diff(lda_model)*), modellen belüli topikhasonlósági mátrixot tudunk vele generálni. A távolság mérésére beállítható mutatók például a *'kullback_leibler' és a 'jaccard'*. A Jaccard távolságnak szüksége van a *num_words* paraméterre is, amivel meg lehet adni, hogy a topikok hány legjellemzőbb szavára számítsa ki az értéket a függvény, így az eredményt nagyban befolyásolja, hogy mekkora szószámot választunk. A Kullback-Leibler Divergence[28] vagy Distance a modell alapján

[26] https://radimrehurek.com/gensim/auto_examples/howtos/run_compare_lda.html (megnyitva: 2024.04.03.)
[27] https://radimrehurek.com/gensim/models/ldamodel.html#gensim.models.ldamodel.LdaModel.diff (megnyitva: 2024.04.03.)
[28] Képlete (Terragni et al., 2021, p. 36.): $KL - DIV(\beta_i, \beta_j) = \sum_{v \in V} \beta_i(V) log \frac{\beta_i(v)}{\beta_j(v)}$



5. ábra: Optimalizált LDA modell topikhasonlósági mátrixa a Kullback-Leibler távolság alapján

kalkulálja a távolságértékeket úgy, hogy összehasonlítja egymással az egyes topikok eloszlását a korpusz szavain (Terragni et al., 2021). A mátrixon a 0 Kullback-Leibler távolság az azonos topikok között látható értelemszerűen. A „Brüsszel & külföld" (6.), valamint a „Kereszténység & siker" (2.) topik között a legkisebb a távolság (0,725), és az is jól látszódik, hogy a „Brüsszel & külföld" (2.) topik minden egyéb topikhoz közel van a többi topiktávolsághoz viszonyítva (0,7 és 0,8 közötti a távolságuk). 0,77 a távolság a „Migráció & krízis" (3.), valamint a „Kereszténység & siker" (2.) topikok között, ez a közelség a topikközi távolságtérképen (*4. ábra*) is látszik. Egymástól a legkülönbözőbbek a „Szerbia & nehézségek" (5.) és a „Hosszútávú & növekedés" (7.) topik, közöttük 1 a távolság mértéke, és a *4. ábrán* is látszik, hogy ezek vannak a legtávolabb egymástól.

Összességében tehát elmondható, hogy az LDA-val generált topikok nem fednek le egyértelműen meghatározható és elkülönülő témaköröket. Vannak közöttük átfedő topikok és jelentésükben távolabb állóak is, de összességében nézve nincsen nagy távolság a topikok között. Az LDA vizualizációs eszközei korlátozottak, más megjelenítési módokhoz (például a dokumentumbeágyazások kétdimenziós klaszterei domináns topik szerint) további kódok szükségesek[29].

[29] https://www.machinelearningplus.com/nlp/topic-modeling-visualization-how-to-present-results-lda-models/ megnyitva: 2024.04.03.)

3.2. AZ LDA-BEÁLLÍTÁSÚ BERTOPIC MODELL EREDMÉNYEI

Az LDA-beállítású BERTopic modell elsődleges különbsége az optimalizált LDA modellhez képest, hogy létrehoz egy outlier topikot is, ugyanis a BERTopic nem kényszerít bele minden dokumentumot egy topikba. A kimaradó dokumentumokat az outlier topikhoz rendeli, így a többi 9 topik változatosabb és koherensebb lehet. Ez a különbség a topikok felcímkézése során nagy könnyebbséget jelentett, így ennél a modellnél sikerült a topikszavak alapján jól körülhatárolható témákat azonosítanom (*8. táblázat*).

Outlier topik		1: Sikeres fejlődés		2: Európa		3: Orosz-ukrán háború		4: Hazafiság	
foreign	0.009	development	0.008	eu	0.019	energy	0.035	freedom	0.026
crisis	0.009	order	0.008	border	0.018	russia	0.033	history	0.016
dont	0.008	success	0.008	member	0.015	ukraine	0.029	live	0.016
development	0.008	look	0.008	migrants	0.015	vo	0.021	homeland	0.013
thing	0.007	still	0.008	party	0.015	gik	0.019	nations	0.013
states	0.007	support	0.008	migration	0.015	russian	0.018	never	0.013
made	0.007	family	0.007	states	0.014	war	0.016	still	0.012
success	0.007	successful	0.007	borders	0.014	agreement	0.016	day	0.012
look	0.007	living	0.007	brussels	0.013	prices	0.015	free	0.012
family	0.007	opportunity	0.007	parliament	0.012	gas	0.015	1956	0.012
5: Covid		6: Egyház		7: Sport		8: Kína		9: Ipar	
weve	0.022	christian	0.044	olympic	0.060	china	0.090	bridge	0.061
dont	0.021	church	0.041	sport	0.053	chinese	0.045	bres	0.044
virus	0.018	christians	0.021	sports	0.038	region	0.029	honourable	0.037
theres	0.018	christianity	0.020	academy	0.032	trade	0.023	sopron	0.034
doctors	0.016	congregation	0.017	games	0.031	growth	0.023	products	0.031
measures	0.016	god	0.016	football	0.028	bank	0.019	szolnok	0.030
theyre	0.015	faith	0.015	budapest	0.023	success	0.017	bonafarm	0.030
defence	0.015	churches	0.014	athletes	0.022	road	0.016	businesses	0.029
pandemic	0.015	immigrant	0.014	olympics	0.021	become	0.015	industry	0.029
healthcare	0.014	reformed	0.013	championships	0.020	sixteen	0.014	project	0.028

8. táblázat: LDA-beállítású BERTopic topikreprezentációi

A topikszavak között előkerültek olyan szótokenek, amelyek a stopszavak eltávolításakor nem voltak az eltávolítandó szavak listájában (tehát nem voltak az *nltk* beépített stopszó listáján, sem az azok eltávolítása után maradó leggyakoribb 95 szó között és nem „.hu”-ra végződőek), és az LDA topikreprezentációiban nem jöttek ki, így nem kerültek külön kiszűrésre. Ezek a szavak az aposztróf nélküli „dont”, „weve”, „theres” és „theyre”, valamint a „vo” és „gik” monogramok (Orbán Viktor és Kiss Gábor István angol szórendű monogramja, interjú felépítésű dokumentumban szerepelnek a monogramjaik a párbeszéd jelölése miatt). Továbbá az „Ipar” (9.) topikban található „bres” tokenen látszódik (eredetileg „Béres”), hogy a BERTopic az átalakítás közben eldobta az ékezetes betűt.

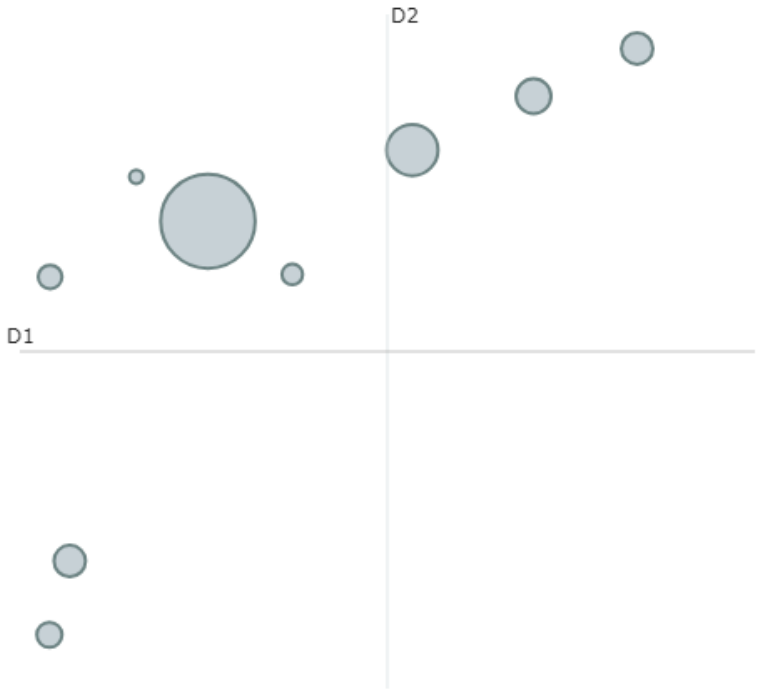
A BERTopic Python implementációjának az LDA-val ellentétben számos beépített vizualizációs lehetősége van. A vizualizációs függvények *custom_labels* paraméterét *True*-ra állítva tudjuk beállítani, hogy az általunk megadott topikneveket jelenítsék meg az ábrákon. A topikneveket a *topic_model.set_topic_labels()* paranccsal tudjuk beállítani szótárként megadva (pl. *{0: "Sikeres fejlődés", 1: "Európa", 2: …}*). Szokás szerint a méretet a *width* és *height* paraméterekkel, a címet pedig a *title* megadásával lehet állítani.

A *topic_model.visualize_documents()*[30] függvénnyel a korpusz dokumentumainak kétdimenziós beágyazásait láthatjuk a domináns topikjuk szerinti klaszterek alapján színezve (*6. ábra*). A paramétereként meg kell adni az illesztésnél is használt dokumentumlistát (*docs*) és az előre generált beágyazásokat (*embeddings*). A parancs egy interaktív ábrát generál, amin

6. ábra: LDA-beállítású BERTopic modell dokumentumainak kétdimenziós klaszterei domináns topik szerint

[30] https://maartengr.github.io/BERTopic/api/plotting/documents.html (megnyitva: 2024.04.04.)

kattintással kijelölhetőek a topikok, amelyek szeretnénk megjeleníteni vagy elrejteni. Szürke színnel az outlier dokumentumok vannak jelölve. A „Sikeres fejlődés” (1.) a legnagyobb és legkiterjedtebb topik, amivel összhangban van az is, hogy a topikszavai kevésbé összefüggőek és a valószínűségük alacsonyabb, mint más topikok jellemző szavainak (*8. táblázat*). Láthatjuk, hogy az „Európa” (2.) és az „Orosz-ukrán háború” (3.) topik közel van a „Sikeres fejlődés” (1.) topikhoz, míg tartalmukban koherensebb témakört fednek le a topikreprezentációjuk alapján. Az „Európa” (2.) topik úgy tűnik magába foglalj a migráció és az Európai Parlament témakörét is, az „Orosz-ukrán háború” (3.) szavai pedig kifejezetten a háborúra és annak következtében tapasztalható válságra vonatkoznak. A „Hazafiság” (4.) topik is közel van a „Sikeres fejlődéshez” (1.), miközben topikszavai egyértelműen a patriotizmusra jellemző kifejezések. Ezek a topikközelségek a topikok együttes előfordulására utalhatnak. Láthatóak kisebb, elkülönülő topikok is, például az „Egyház” (6.), „Sport” (7.) és „Kína” (8.).



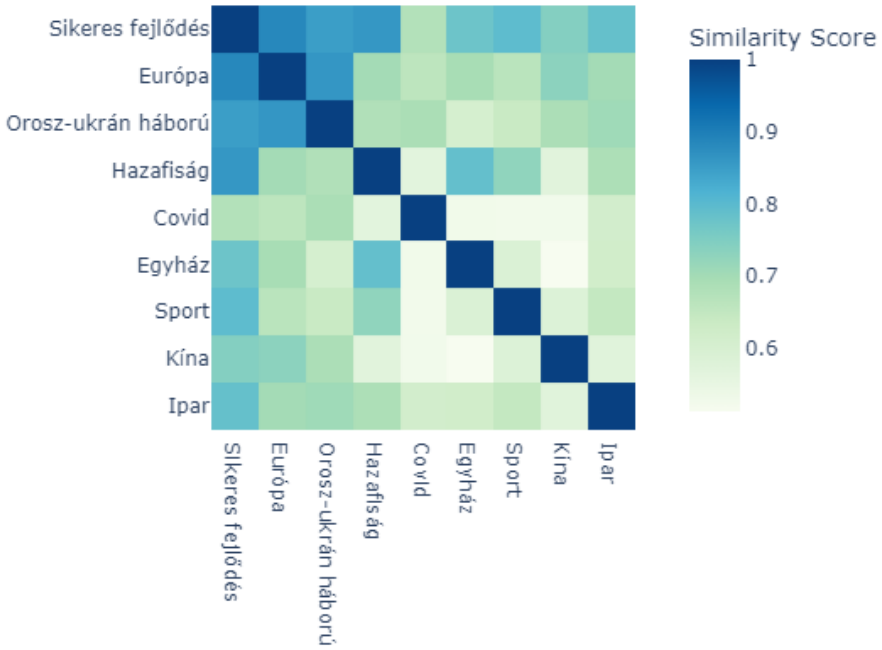
7. ábra: LDA-beállítású BERTopic modell topikközi távolságtérképe

A *topic_model.visualize_topics()*[31] parancs az UMAP segítségével két dimenzióra vetíti a c-TF-IDF reprezentációkat, és így mutatja be a topikreprezentációk távolságát egymástól (*7. ábra*). Ez a vizualizáció megfelel a *pyLDAvis* topikközi távolságtérképének, azzal a különbséggel, hogy

[31] https://maartengr.github.io/BERTopic/api/plotting/topics.html (megnyitva: 2024.04.04.)

mivel a BERTopic a topikgeneráláskor egy dokumentumot egy topikhoz rendel, és ezután lehet a dokumentumokra kiszámoltatni, hogy mely topikok keverékeként állnak elő. Így a topikok mérete nem korpuszban az összes előfordulásuk gyakorisága, hanem azon dokumentumok száma, amelyekben domináns topikként szerepelnek. Láthatunk egy kiugróan nagy méretű topikot, a „Sikeres fejlődést" (1.), amely a *6. ábrán* is a legnagyobb topik volt. Ehhez közel helyezkednek el a legkisebb topikok, az „Ipar" (9.), „Kína" (8.) és a „Sport" (7.). A *7. ábrán* azt látjuk, hogy a „Hazafiság" (4.) és „Egyház" (6.) topikok egymáshoz közel és némileg távolabb helyezkednek el a többi topiktól (az ábra bal alsó részén), míg a *6. ábrán* nem látszódott nagy leszakadás ezek esetében. Látható, hogy nincsenek átfedések a topikok között, ami abból is adódik, hogy az egymáshoz közel lévő topikok mérete nem olyan nagy, hogy átfedésbe kerüljenek a topikközi távolságtérképen.

A *topic_model.visualize_heatmap()*[32] parancs a koszinusz hasonlóságot (cosine similarity) mutatja be a topikok között, amely a topikreprezentációk által bezárt szög koszinusza a vektortérben. Az LDA topikhasonlósági mátrixával tehát nem összevethetőek az itt látható értékek, ezt az eltérő színskálával is igyekeztem jelezni. Az 1-es érték a teljes azonosságot, míg a 0-hoz közeli értékek a különbözőséget jelölik. A hőtérképről (*8. ábra*) leolvasható, hogy a

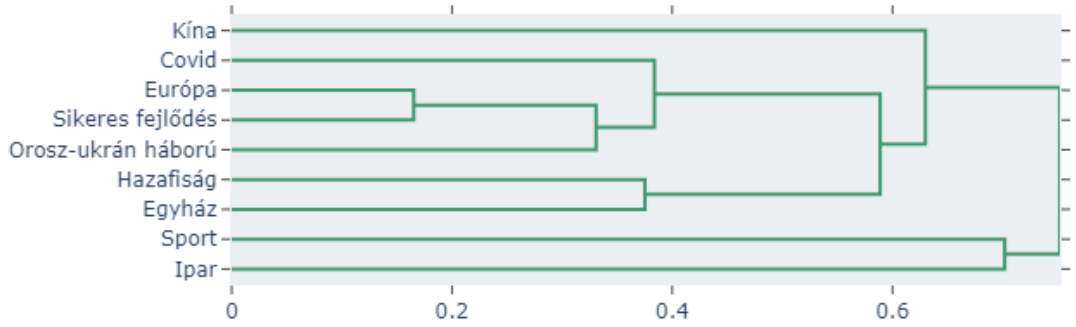


8. ábra: LDA-beállítású BERTopic modell topikhasonlósági mátrixa a koszinusz hasonlóság alapján

[32] https://maartengr.github.io/BERTopic/api/plotting/heatmap.html (megnyitva 2024.04.04.)

„Sikeres fejlődés” (1.) topik a „Covid” (5.) topikon kívül erős hasonlóságot mutat a többi topikkal (0,74 és 0,88 közötti koszinusz hasonlóság). Az „Európa” (2.) és „Orosz-ukrán háború” (3.) topikok között is erős a hasonlóság (0,86), ezek a topikok a *6. ábrán* is egymás mellett helyezkedtek el. A topikhasonlósági mátrix szerint az egymáshoz a legkisebb mértékű hasonlóság a „Kína” (8.) és „Egyház” (6.) topikok között van (0,51), ezt nem sokkal haladja meg a „Covid” (5.) és a „Egyház” (6.), „Sport” (7.) és „Kína” (8.) közötti távolság (0,53). Összességében is elmondható, hogy a „Covid” (5.) topik a legkülönbözőbb topik.

A *topic_model.visualize_hierarchy()*[33] paranccsal dendogramszerűen láthatjuk a topikjaink hierarchikus struktúráját. A függvény a topikokat a Ward-féle eljárással vonja össze a koszinusz távolságok alapján, tehát azokat topikokat vonja össze, melyek összevonásával a legkisebb lesz a belső szórásnégyzet növekedése. Arra következtethetünk belőle, hogyan vonódnának össze a topikjaink, ha csökkenne a topikszám. Azt láthatjuk a dendogramon (*9. ábra*), hogy a „Sikeres fejlődés” (1.) topik először az „Európát” (2.), majd az „Orosz-ukrán háborút” (3.) és a „Covid” (5.) topikot kebelezné be. A „Hazafiság” (4.) és „Egyház” (6.) topik összeolvadnának, mielőtt csatlakoznak a többi topik együtteséhez. Legtovább a „Sport” (7.) és „Ipar” (9.) topikok kombinációja maradna különálló topik.



9. ábra: LDA-beállítású BERTopic modell topikjainak hierarchikus sturktúrája

Összességében változatos, könnyen körülhatárolható topikokat generált az LDA-beállítású BERTopic modell, amelyek között ugyan van hasonlóság, de nem redundánsak. Az optimalizált LDA modellhez képesti legnagyobb különbsége az outlier topik megengedése, ami hatalmas mértékben javította a többi 9 topik koherenciáját és diverzitását.

[33] https://maartengr.github.io/BERTopic/api/plotting/hierarchy.html (megnyitva: 2024.04.04.)

3.3. AZ OPTIMALIZÁLT BERTOPIC MODELL EREDMÉNYEI

Az optimalizált BERTopic modell 16+1 topikreprezentációinak szavait és szóvalószínűségeit a *9. táblázat* mutatja be. A táblázatban az általam adott topiknevek láthatóak a topikokhoz tartozó 10-10 legvalószínűbb szóval és valószínűségükkel. Ennél a modellnél nem távolítottam el

Outlier topik		1: Általános		2: Covid		3: Törökország & Egyiptom		4. Egyház	
one	0.192	hungary	0.203	virus	0.274	egypt	0.465	church	0.403
european	0.192	hungarian	0.199	doctors	0.261	turkish	0.453	christian	0.351
us	0.191	european	0.197	healthcare	0.245	turkey	0.440	christians	0.312
hungary	0.190	us	0.192	vaccine	0.242	egyptian	0.342	congregation	0.299
hungarian	0.189	europe	0.190	pandemic	0.241	president	0.339	christianity	0.285
people	0.189	gentlemen	0.189	measures	0.239	excellency	0.297	churches	0.262
europe	0.188	one	0.186	hospital	0.229	cooperation	0.289	reformed	0.257
years	0.183	people	0.184	defence	0.225	business	0.281	immigrant	0.248
country	0.176	ladies	0.182	restrictions	0.224	relations	0.250	persecution	0.243
minister	0.175	hungarians	0.180	infection	0.213	normalchar	0.249	faith	0.239

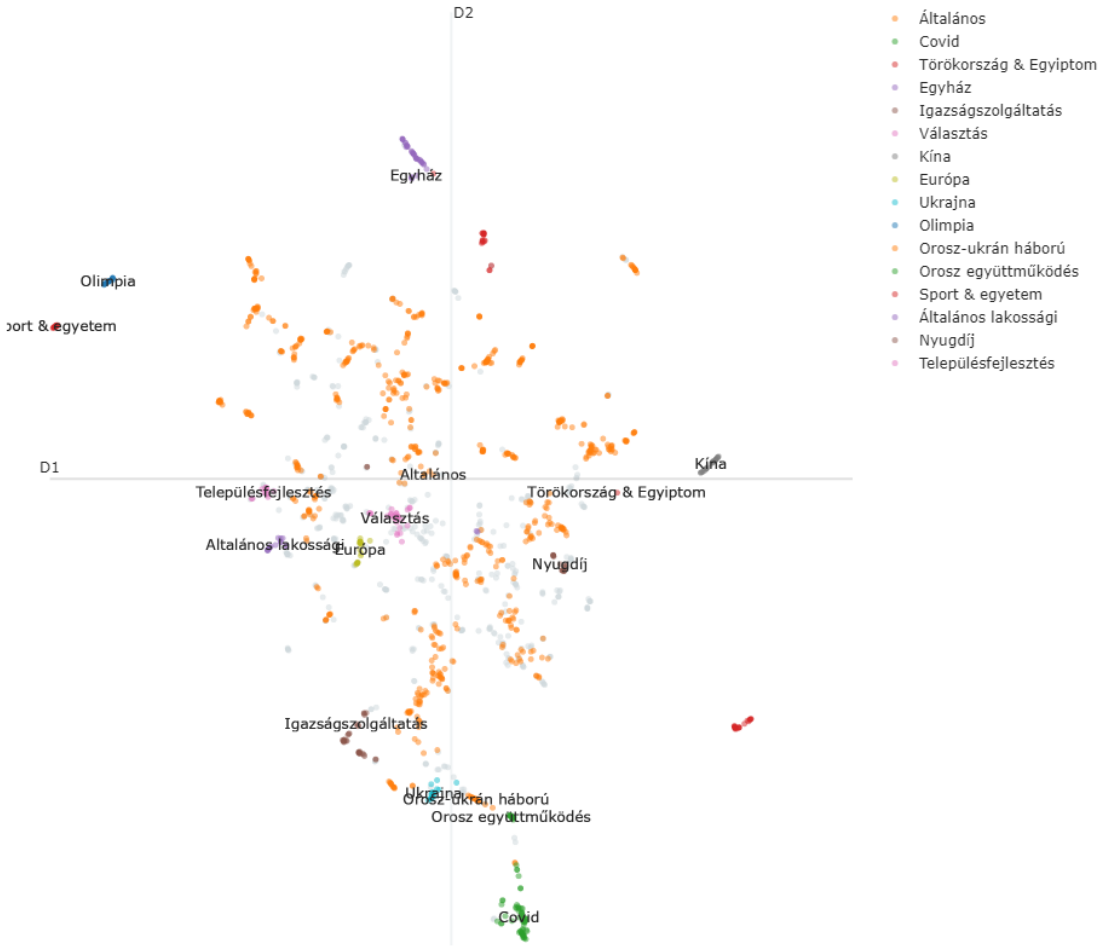
5: Igazságszolgáltatás		6. Választás		7: Kína		8: Európa	
law	0.437	election	0.249	china	0.571	class	0.317
oath	0.421	elections	0.227	chinese	0.432	liberal	0.233
service	0.388	party	0.226	central	0.374	christian	0.226
police	0.382	people	0.222	region	0.301	europe	0.219
enforcement	0.346	want	0.220	cooperation	0.283	us	0.209
prosecution	0.328	opposition	0.210	trade	0.274	political	0.204
takers	0.322	fidesz	0.210	bank	0.267	democracy	0.204
officers	0.314	hungarian	0.206	sixteen	0.262	years	0.202
chief	0.302	right	0.201	belt	0.260	european	0.199
prosecutor	0.299	liberal	0.200	lmfalussy	0.258	say	0.197

9: Ukrajna		10: Olimpia		11: Orosz-ukrán háború		12: Orosz együttműködés	
ukraine	0.513	olympic	0.673	energy	0.444	russia	0.600
ukrainian	0.372	games	0.505	sanctions	0.444	russian	0.498
transcarpathia	0.317	olympics	0.427	russia	0.433	cooperation	0.463
ukraines	0.270	championships	0.412	gas	0.417	president	0.395
percent	0.258	sport	0.411	prices	0.396	putin	0.382
prices	0.256	sports	0.410	russian	0.353	energy	0.379
ukrainians	0.256	rio	0.385	price	0.339	brazil	0.363
utility	0.253	athletes	0.358	oil	0.324	relations	0.351
weekend	0.245	medals	0.354	war	0.319	nuclear	0.335
living	0.235	committee	0.319	2023	0.317	agreement	0.325

13: Sport & Egyetem		14: Általános lakossági		15: Nyugdíj		16: Településfejlesztés	
academy	0.501	oecd	0.258	year	0.342	cegld	0.328
sport	0.480	civic	0.242	pensioners	0.334	bres	0.297
football	0.480	years	0.228	supplement	0.324	pharmaceuticals	0.274
basketball	0.470	tax	0.227	pension	0.317	villages	0.243
pusks	0.412	ladies	0.218	per	0.314	people	0.243
academies	0.412	time	0.217	cent	0.307	village	0.239
team	0.363	something	0.215	2016	0.297	mayor	0.236
sports	0.344	say	0.214	pensions	0.264	city	0.232
rtgber	0.314	gentlemen	0.212	budget	0.260	development	0.230
honvd	0.312	rvsz	0.211	2017	0.259	kindergarten	0.227

9. táblázat: Optimalizált BERTopic topikreprezentációi

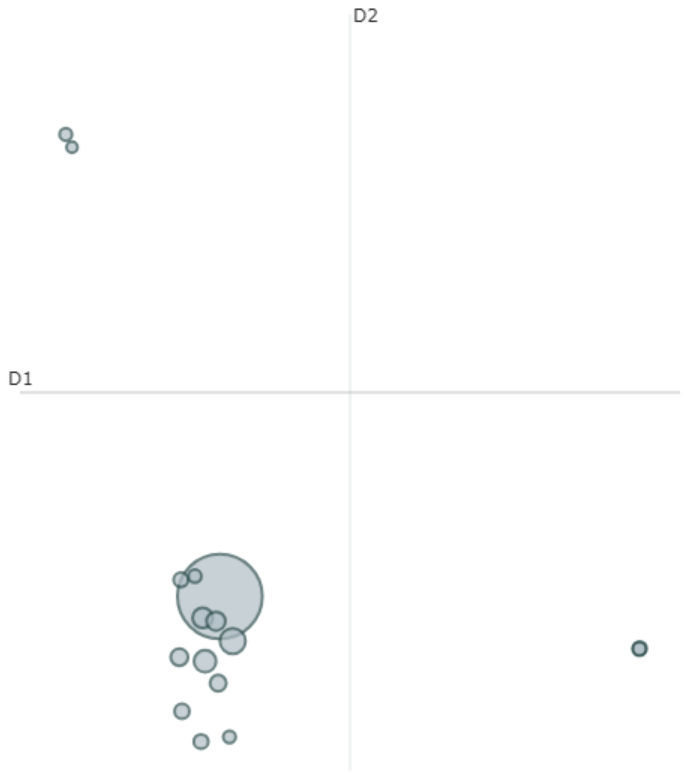
manuálisan a gyakori szavakat, hanem a c-TF-IDF modell beállításával csökkentettem az előfordulásukat, így látható, hogy az outlier topikon túl az „Általános” (1.) topikban ezek a kiugróan magas gyakoriságú szavak jelentek meg. Az LDA-beállítású BERTopic modellnél még felmerülő stopszavakat (pl. „dont”, „weve”, „vo”, „gik”) itt már kiszűrtem a stopszavak eltávolítása lépésben, viszont szintén látszódik például a „cegld” (Cegléd), „pusks” (Puskás), „honvd” (honvéd) szótokeneknél, hogy a BERTopic a topikgenerálás során eldobja az ékezetes betűket. Az optimalizált LDA modellhez képest jóval könnyebb volt a topikok felcímkézése, de az LDA-beállítású BERTopic modellhez képest némileg nehezebb, például a 14. topik szavait nem találtam koherensnek. Ugyanakkor több különböző és jól körülhatárolható téma is megjelent itt, ami a másik két modellben nem (pl. „Törökország & Egyiptom” (3.) és „Nyugdíj” (15.)). A topikszavakból is látható, hogy vannak átfedőbb topikok, a „people”, „hungarian”, „european”, „cooperation” szavak például több topikban is szerepelnek.



10. ábra: Az optimalizált BERTopic modell dokumentumainak kétdimenziós klaszterei domináns topik szerint

Az optimalizált BERTopic modell dokumentumklasztereinek ábrája (*10. ábra*) zsúfolt és nehezebben olvasható, mint az LDA-beállítású BERTopic modellé (*6. ábra*), de az összehasonlítás érdekében fontos látni ezt is. Jól látszik, hogy az „Általános” (1.) topik többnyire megfeleltethető az ottani „Sikeres fejlődés” topiknak, míg az „Európa” (8.) sokkal kisebb méretű klasztert képez, mint a másik modell azonos nevű klasztere. Az „Orosz-ukrán háború” (11.), „Orosz együttműködés” (12.) és „Ukrajna” (9.) topikok megfeleltethetőek a másik modell „Orosz-ukrán háború” topikjának, összesítve hozzávetőlegesen azonos méretűek. A „Covid” (2.) és „Egyház” (4.) topikok megfeleltethetőnek látszódnak a másik modell azonos nevű topikjaival.

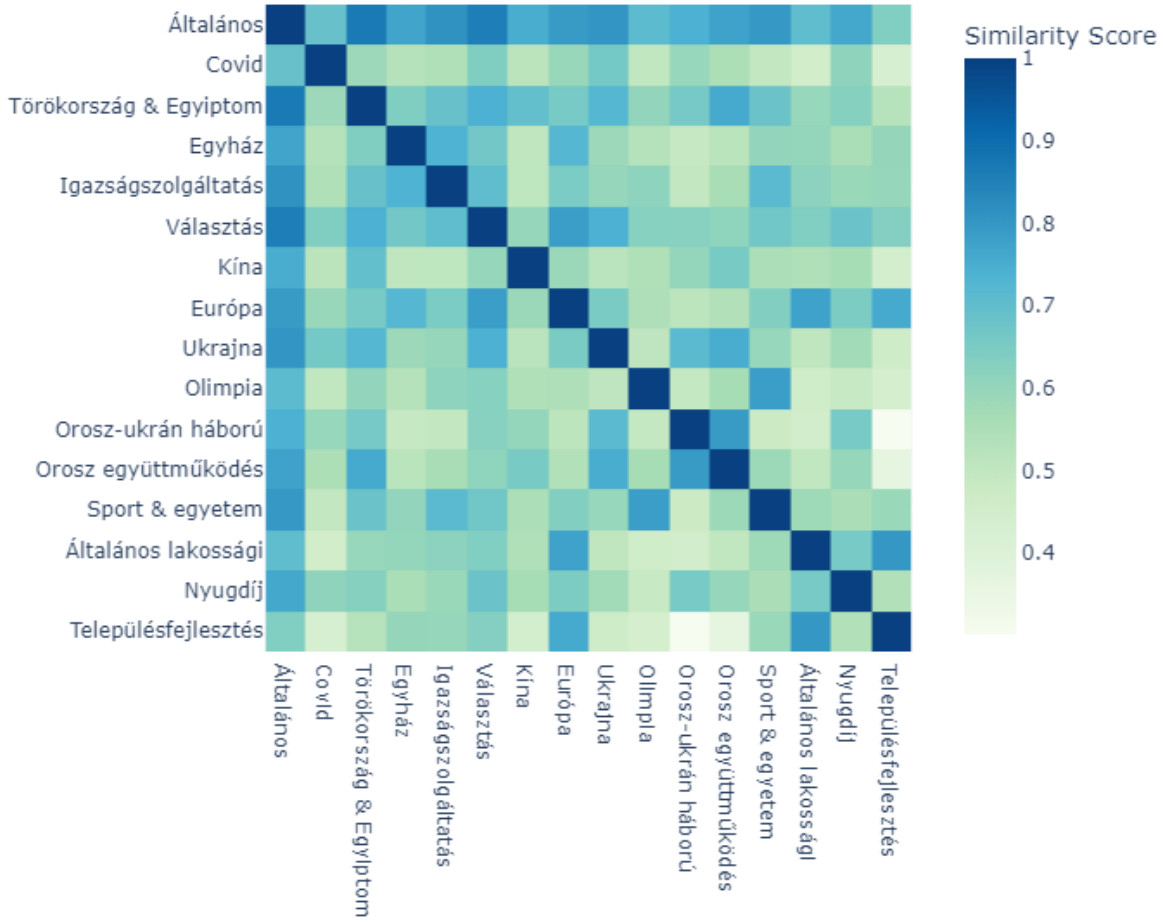
Az UMAP segítségével két dimenzióra vetített topikközi távolságtérképpel kapcsolatban (*11. ábra*) ismét kiemelendő, hogy a méretüket az határozza meg, hány dokumentumban jelennek meg domináns topikként, nem pedig az összesített előfordulási gyakoriságuk a korpuszban, mint az LDA esetében. Jól látható egy kiugróan nagy méretű topik, az „Általános” (1.), és 11 további apró topik körülötte. Az is megfigyelhető, hogy több kisebb topik az „Általános” (1.)



11. ábra: Az optimalizált BERTopic modell topikközi távolságtérképe

topik területén helyezkedik el, tehát jelentésükben átfedőek ezek a topikok. Négy topik van, ami ettől a csomóponttól nagy távolságra van, tehát meglehetősen különböző jelentésűek. A jobb alsó sarokban egymáson helyezkedik el az „Orosz-ukrán háború” (11.) és „Orosz együttműködés” (12.) topik, míg a bal felső sarokban egymáshoz közel az „Általános lakossági” (13.) és a „Településfejlesztés” (15.) topik. Tehát azok a többi topiktól jelentésükben eltérőek, de egymáshoz hasonlóak.

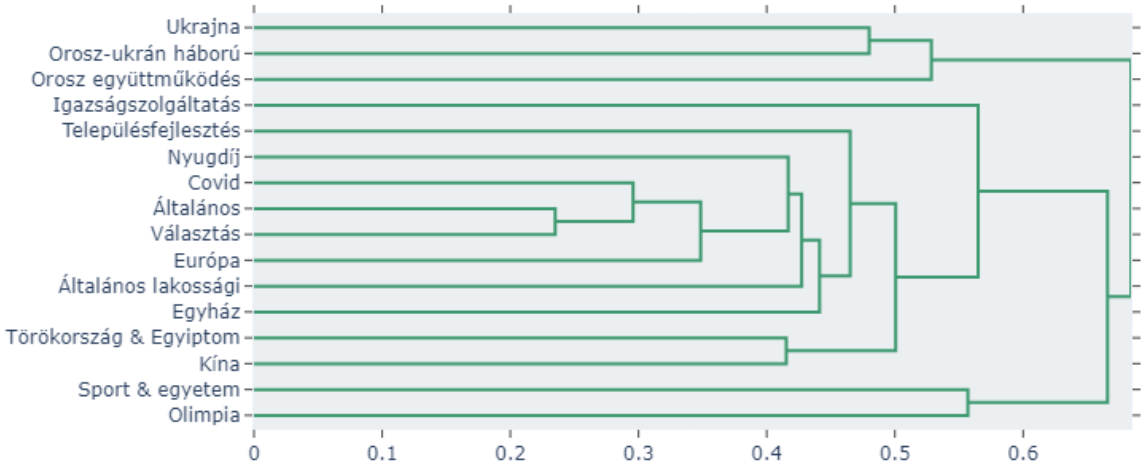
A topikok egymáshoz való hasonlóságát vagy különbözőségét a koszinusz hasonlóság alapján generált mátrix mutatja be (*12. ábra*). Jól látható a hőtérképen, hogy az az „Általános” (1.) topik és más topikok között az átlagosnál nagyobb mértékű a hasonlóság (0,64 és 0,87 közötti). Látható továbbá, hogy a „Sport & egyetem” (13.) topik és az „Olimpia” (10.) között magas a hasonlóság mértéke (0,78), ahogy az „Európa” (8.), „Általános lakossági” (14.) és a „Településfejlesztés” (16.) topikok között is (0,76 és 0,8 közötti). Szintén magas koszinusz



12. ábra: Az optimalizált BERTopic modell topikhasonlósági mátrixa a koszinusz hasonlóság alapján

hasonlóság van „Európa” (13.) és a „Választás” (6.) topik között (0,78). A legkevésbé hasonló topikok az „Orosz-ukrán háború” (11.) és a „Településfejlesztés” (16.), valamint utóbbi az „Orosz együttműködés” (12.) topikkal (0,3 és 0,4).

A topikok hierarchikus struktúrájánál láthatjuk, milyen sorrendben olvadnának össze a topikok a topikszám csökkentésével. (*13. ábra*). Először a „Választás” (6.) topik olvadna az „Általánosba” (1.), majd a „Covid” (2.), az „Európa” (8.) és a „Nyugdíj” (15.). Utoljára a „Sport & egyetem” (13.) és „Olimpia” (10.) topikok kombinációja, valamint az „Ukrajna” (9.), „Orosz-ukrán háború” (11.) és „Orosz együttműködés” (12.) topikhármas olvadna be.



13. ábra: Az optimalizált BERTopic modell topikjainak hierarchikus struktúrája

Összességében elmondható, hogy az optimalizált BERTopic modellel készült topikok koherensek és széttagoltabbak, mint az LDA-beállítású BERTopic modell eredményei. Ezáltal több témakör került megjelenítésre, ami a másik modellnél „rejtve” maradt. Ugyanakkor a gyakori szavak eltávolítása nélkül az „Általános” (1.) topik túl nagy súlyt kapott a többi topikhoz képest, bár tekinthetünk rá úgy is, hogy reális képet mutat a beszédek tartalmáról attól függetlenül, hogy tartalmi szempontból nem hordoz jól körülhatárolható jelentést.

3.4. A MODELLEK EREDMÉNYEINEK ÖSSZEHASONLÍTÓ ELEMZÉSE

A három modell összehasonlításának kvantitatív szempontjait a *10. táblázat* mutatja be. Az LDA és az LDA-beállítású BERTopic modell közötti különbséget a topikgenerálási eljáráson túl az outlier topik megengedése jelenti, ami az LDA-nál nem lehetséges, a BERTopic-nál pedig a modell működéséből adódik. A korábbi alfejezetekben bemutatásra került topikreprezentációk alapján láthattuk, hogy ez sokkal koherensebb és változatosabb topikokat eredményezett, amit a coherence score és a topic diversity mutatók is jól jellemeznek[34]. Az LDA modell az optimalizálás során bármely beállítással negatív coherence scoret generált, bár láthatjuk, hogy a BERTopic-ok esetén is 0 közelében maradt a mutató, ami a korpusz sajátossága lehet[35]. Az optimalizált BERTopic-nál a topikszám és a stopszó lista is eltér, ennél a modellnél a Schofield és társai (2017) által ajánlott módszert, az output alapú stopszólista készítést alkalmaztam. Coherence score alapján ez a modell teljesített a legjobban, míg topic diversity szempontjából az LDA-beállítású BERTopic modell volt a legjobb.

Modell	Topikszám	Stopszó lista	Coherence score	Topic diversity
Optimalizált **LDA**	9	- nltk alapértelmezett stopszavai - ezeken kívüli 95 leggyakoribb szó - „.hu” végződésű szótokenek	-0,014	0,576
LDA-beállítású **BERTopic**	9 + outlier	- nltk alapértelmezett stopszavai - ezeken kívüli 95 leggyakoribb szó - „.hu” végződésű szótokenek	0,033	0,858
Optimalizált **BERTopic**	16 + outlier	- nltk alapértelmezett stopszavai - utólag meghatározott lista a topikreprezentációk szavai alapján	0,087	0,773

10. táblázat: A három modell és teljesítményük bemutatása

Kvalitatív megközelítésből vizsgálva a három modellt, az LDA topikreprezentációi voltak a legkevésbé értelmezhetőek, miközben a BERTopic modellek emberi szemmel is egy-egy összefüggő és könnyen meghatározható témakört fedtek le. Coherence score szempontjából a 16+1 topikszám teljesített a legjobban, de így különváltak szorosan összefüggő topikok is, például „Orosz-ukrán háború”, „Orosz együttműködés” és „Ukrajna”, amik korábban egy topikot képeztek.

[34] A mutatók működése és értéktartománya a 2. fejezetben került bemutatásra.

[35] Grootendorst 2022-es tanulmányában 3 különböző korpuszon vizsgálta a topikmodelleket, és a Donald Trump twitter bejegyzéseit tartalmazó korpuszon az LDA modell coherence score-ja -0,011 volt, a BERTopic modellnek pedig 0,066, és így is utóbbi teljesített a legjobban a hat vizsgált modell közül.

Optimalizált BERTopic modell topikjai	LDA-beállítású BERTopic modell leghasonlóbb topikjai	Koszinusz hasonlóság
Outlier topik	Outlier topik	0.997
1: Általános	1: Sikeres fejlődés	0.991
2: Covid	5: Covid	1.000
3: Törökország & Egyiptom	1: Sikeres fejlődés	0.884
4: Egyház	6: Egyház	1.000
5: Igazságszolgáltatás	1: Sikeres fejlődés	0.836
6: Választás	1: Sikeres fejlődés	0.870
7: Kína	8: Kína	0.999
8: Európa	Outlier topik	0.834
9: Ukrajna	3: Orosz-ukrán háború	0.910
10: Olimpia	7: Sport	0.959
11: Orosz-ukrán háború	3: Orosz-ukrán háború	0.893
12: Orosz együttműködés	3: Orosz-ukrán háború	0.917
13: Sport & Egyetem	7: Sport	0.927
14: Általános lakossági	Outlier topik	0.779
15: Nyugdíj	Outlier topik	0.792
16: Településfejlesztés	Outlier topik	0.706

11. táblázat: Optimalizált és LDA-beállítású BERTopic modell topikjainak összevetése I.

Ugyanakkor megjelent a „Nyugdíj”, „Igazságszolgáltatás” és „Törökország & Egyiptom”, amely témakörök az LDA-beállítású modell topikreprezentációiban egyáltalán nem látszottak. A *11. táblázatban* jól látszik, hogy előbbi az Outlier topikból válhatott ki, utóbbi kettő pedig a „Sikeres fejlődés topikból”, ugyanis ahhoz vannak a legközelebb a koszinusz hasonlóság alapján. Az LDA-beállítású modell „Sport” topikja szétvált „Sport & Egyetem” és „Olimpia” topikokra. Az optimalizált BERTopic modellben négy topik is van, ami az LDA-beállítású BERTopic Outlier topikjához van a legközelebb. A két BERTopic modellben a „Covid”, „Egyház” és „Kína” topikok szinte teljesen megegyeznek.

LDA-beállítású BERTopic modell topikjai	Optimalizált BERTopic modell leghasonlóbb topikjai	Koszinusz hasonlóság
Outlier topik	Outlier topik	0.997
1: Sikeres fejlődés	1: Általános	0.991
2: Európa	Outlier topik	0.938
3: Orosz-ukrán háború	12: Orosz együttműködés	0.917
4: Hazafiság	1: Általános	0.859
5: Covid	2: Covid	1.000
6: Egyház	4: Egyház	1.000
7: Sport	10: Olimpia	0.959
8: Kína	7: Kína	0.999
9: Ipar	1: Általános	0.789

12. táblázat: Optimalizált és LDA-beállítású BERTopic modell topikjainak összevetése II.

A *12. táblázat* alapján látszik, hogy az LDA-beállítású BERTopic „Hazafiság” topikja az optimalizált BERTopic „Általános” topikjában van, tehát az eltérő paraméterezés, és főként a különböző stopszavazás miatt ez a témakör „elveszett” az optimalizálás során. Továbbá az „Ipar” topik az optimalizált modell „Általános” topikjába és az „Európa” topik az Outlier topikjába olvadt be.

A BERTopic modellek összehasonlítását a topikbeágyazásuk alapján tudtam elvégezni a *sklearn* csomag *cosine_similarity* funkciójával[36], míg a modellük eltérő működése miatt az LDA és BERTopic topikjainak összehasonlítását nem tudtam ugyanezzel a metrikával mérni. Így helyette az azonos szótokeneik arányának meghatározásával mértem a topikhasonlóságukat. Ehhez a legvalószínűbb 25 topikszavakat vizsgáltam, hasonlóan az alkalmazott topic diversity mutatóhoz. Az eltérő számítási metódus miatt fontos figyelembe venni, hogy nem összevethető a két mutató értékének nagysága, csak önmagukban értelmezendőek a modellek közötti leghasonlóbb topikok megtalálásához.

A *13. táblázatban* jól látszik annak a hatása, hogy a BERTopic megengedi az outlierek létrehozását, ezáltal az optimalizált LDA modellben megtalált topikok nagy része az LDA-beállítású BERTopic modell Outlier topikjába került. Ezeken túl a „Támogatás & EU” topik az „Európa” topikra, az „Energia & fejlődés” a „Sikeres fejlődés” topikra hasonlított leginkább, de ezeknek is csak rendre 16% és 20% az átfedésük, ami alacsonynak számít, ha azt vizsgáljuk, mennyire sikerült azonos topikokat generálnunk.

Optimalizált LDA modell topikjai	LDA-beállítású modell leghasonlóbb topikjai	Azonos szótokenek százaléka
1: Támogatás & EU	2: Európa	16%
2: Kereszténység & siker	Outlier topik	22%
3: Migráció & krízis	Outlier topik	18%
4: Támogatás & választás	Outlier topik	22%
5: Szerbia & nehézség	Outlier topik	22%
6: Brüsszel & külföld	Outlier topik	22%
7: Hosszútávú & növekedés	Outlier topik	26%
8: Energia & fejlődés	1: Sikeres fejlődés	20%
9: Nyugat & problémák	Outlier topik	20%

13. táblázat: Optimalizált LDA és LDA-beállítású BERTopic modell topikjainak összevetése

[36] https://maartengr.github.io/BERTopic/getting_started/tips_and_tricks/tips_and_tricks.html#finding-similar-topics-between-models (Megnyitva: 2024.04.09.)

Megállapítható, hogy a vizsgált korpuszon az LDA nem teljesített jól, redundáns és nem összefüggő topikreprezentációkat hozott létre, amelyek a stopszavazás optimalizálását követően is általánosak voltak és számos funkciószót tartalmaztak. Az LDA-beállítású BERTopic modell eredményei nagy mértékben eltérőek lettek, aminek fő oka HDBSCAN klaszterező lépéséből adódó outlier topik generálása. Ez mind a topikok értelmezhetőségére, a topikkoherenciára, mind a topikdiverzitásra jó hatással volt. A topikreprezentációkat elolvasva már összefüggő és magas információtartalmú témaköröket láthatunk. A BERTopic modell optimalizálásával közel kétszeresére nőtt a topikszám, ezáltal számos új témakör is láthatóvá vált. Ezek közül leginkább azok voltak értékesek, amelyek általános témájú topikból kiváló kisebb és konkrétabb témakört reprezentáltak. Az eltérő stopszavazás miatt olyan topikok is voltak, amelyek az LDA-beállítású modellben még jelen voltak, de a BERTopic modellben nem kerültek elő.

ÖSSZEGZÉS

Dolgozatomban egy új topikmodellezési technika, a BERTopic működését és teljesítményét mutattam be az elterjed LDA modellel szemben. A gyakorlati összehasonlításhoz egy LDA és két BERTopic modellt vizsgáltam Orbán Viktor angol nyelvű miniszterelnöki beszédeinek korpuszán. Az optimalizált LDA modellnél meghatározott beállításokat alkalmaztam az egyik BERTopic modellen, és optimalizált beállításokat a másikon. A modellek kiértékeléséhez topikkoherencia és topikdiverzitás mutatókat, valamint a topikreprezentációk értelmezhetőséget vizsgáltam.

Az optimalizált LDA modell redundáns és nem összefüggő topikokat eredményezett, míg mindkét BERTopic modell változatos, koherens és specifikus topikokat hozott létre. A különbségnek több oka is volt. A BERTopic megőrzi a dokumentumon belüli szemantikai kapcsolatokat, szemben az LDA által alkalmazott szózsák modellel, így a topikreprezentációi összefüggőbbek. Továbbá a klaszterező lépésben használt HDBSCAN által létrejön egy outlier topik is, így a többi topik egy-egy jól körülhatárolt és az általánosnál specifikusabb, érdekesebb témakört jelölhet, ahogy Egger és Yu (2022) is találták a topikmodelleket összehasonlító kutatásukban.

Gyakorlati előnye a BERTopic-nak, hogy alkalmazása a Python implementációval rendkívül egyszerű, miközben számos finomhangolási lehetősége van, és moduláris felépítésének köszönhetően a modell rugalmasan igazítható az aktuális kutatáshoz. Az egyes lépéseiben alkalmazott eljárások helyett tetszőlegesen alkalmazhatóak más technikák, kevés megkötéssel, így a modell mindig a természetesnyelv-feldolgozás legkorszerűbb megoldásait tudja alkalmazni, ahogy arra Grootendorst (2022) is rámutat. Az eljárások kicserélése ráadásul az úgynevezett pipeline paraméterek megadásával egyszerűen megtehető, ahogyan azt az 2.3. alfejezetben is bemutattam.

A BERTopic praktikusságát erősíti, hogy az LDA-val ellentétben nem szükséges a vizsgált korpusz előfeldolgozása, ami idő- és munkaigényes folyamat. A modell modularitása viszont azt is lehetővé teszi, hogy a dokumentumbeágyazás és a modellillesztés során eltérő előfeldolgozású korpuszt használjunk, aminek az előnyét Grootendorst (2022) is hangsúlyozza. A dokumentumok beágyazását ajánlott az eredeti dokumentumokon elvégezni a szemantikai

kapcsolatok pontos megőrzéséhez, ugyanakkor az illesztés során megadhatunk tokenizált, előfeldolgozott, stopszavak eltávolítása utáni korpuszt is, ha a kutatásunkhoz ez szükséges. Ennek kihasználásával tudtam elkészíteni az LDA-beállítású BERTopic modellt.

A korpusz-specifikus stopszavak meghatározása jelentette a legnagyobb kihívást az optimális LDA modell megtalálásához. A BERTopic esetében bár van az előzetes stopszó eltávolításra lehetőség, nem szükséges lépés, ugyanis több egyéb módot is kínál rá az implementáció, amelyekkel kiszűrhetőek a topikreprezentációkból a megadott szavak, vagy csökkenthető az előfordulása a gyakori szavaknak. Az optimalizált beállítású BERTopic modellben láthattuk, hogy ezek kombinációja alkalmasabb a szövegek eredeti tartalmának visszaadásához, hiszen nem kellett eldobni a leggyakoribb szavakat sem (pl. „hungary", „hungarians"), mégsem torzították el az összes topikot, mint az LDA stopszavazás előtti verziójában. Ezáltal és a magasabb topikszámnak köszönhetően számos érdekes topik megjelent az optimalizált BERTopic modellben, mint az LDA-beállításúban.

A BERTopic az LDA-val szemben képes automatikus topikszám megtalálásra. Ahogyan Egger és Yu (2022) is rámutat, ez többnyire túl magas topikszámot eredményez, de a klaszterek minimum méretének növelésével optimalizálható. Így viszont nem érvényesül az automatikus topikszám meghatározásának előnye, hiszen szükséges manuális beavatkozás és tesztelés.

Az outlierek megengedését a modell előnyei közé sorolom, a korábban is említett topikkoherencia és topikdiverzitás javulása miatt. Ellenben Egger és Yu (2022) a modell hátrányai között jelölte meg, hogy nagy számú outliert generál, amely valóban helytálló. Ha a kutatáshoz szükséges, akkor az outlier dokumentumok belekényszeríthetőek egy-egy klaszterbe egy egyszerű paranccsal (részletek a 2.3. fejezetben).

A BERTopic korlátai közé tartozik, hogy alapvetően nem a topikok keverékeként értelmezi a dokumentumokat, hanem a belőlük alkotott klasztereket feleleti meg a topikoknak. Ezután ugyan a HDBSCAN valószínűségi mátrixa alkalmazható a dokumentumbeli topikkeveredés meghatározására (Grootendorst, 2022), de a számos vizualizációs lehetőség mind a dokumentumok klasztereiből alkotott topikokra vonatkozik.

A BERTopic további hátránya, hogy a topikreprezentációk szavai a beágyazott dokumentumok klasztereiből szózsák modellel kerülnek kiválogatásra (Grootendorst, 2022), ami által a topik szavai között több hasonló szótoken lehet (pl. egyes szám, többes szám, más szófajú alak). Ha

a kutatásban a tartalmi elemzés van a fókuszban, akkor cél lehet a topikhoz tartozó változatosabb szókincs megtalálása is, amelyeknek magasabb a topikhoz hozzáadott információjuk értéke.

A BERTopic további fejlesztési lehetősége beépített metrikák hozzáadása, amelyekkel a modellek jóságát lehet mérni, amelynek hiányára Egger és Yu (2022) is rámutat. Kutatásom során a topic diversity mutatót magam konstruáltam Dieng és társai 2020-es tanulmánya alapján, a coherence score mutatót pedig a *gensim* csomagból használtam, de az eltérő implementáció miatt alkalmazása körülményes volt. Továbbá hiányzik a más topikmodellek implementációjával megegyező mutatók alkalmazásának lehetősége (például a BERTopic rendre koszinusz hasonlóságot vizsgál a topikjai között, míg az LDA-ban négy másik hasonlósági mutató alkalmazható, de a koszinusz hasonlóság nem).

Kutatásom korlátja, hogy egyetlen korpuszon vetettem össze a modelleket, így a korpusz sajátosságai is befolyásolhatták a teljesítményükről alkotott következtetéseimet. Így viszont a paraméterek beállítási lehetőségére helyezhettem a hangsúlyt, és a célkitűzéseim között megjelölt gyakorlati útmutatóként is szolgáló dolgozatot írhattam. Értékes tapasztalatokat dokumentálhattam részletesen, amelyek későbbi kutatások, elemzések során hasznosak lehetnek.

Továbbá a kutatásom során Orbán Viktor angol nyelvű miniszterelnöki beszédeinek tartalmi elemzése nem volt célom. Bár a BERTopic modellek jól teljesítettek a beszédek látens topikjainak feltárásában, fontos figyelembe venni, hogy a topikok felcímkézése a szubjektív kutatói döntésem alapján történt, más kutató feltehetően más elnevezéseket használt volna. A témában rejlő további értékes kutatás lenne a beszédek szakszerű tartalmi elemzése, alkalmazva a BERTopic dinamikus topikmodellezési lehetőségét, amivel az időmenzió bevonásával a beszédek témaköreinek változását is fel lehetne tárni, összehasonlítva a hazai és világszintű eseményekkel.

IRODALOMJEGYZÉK

Angelov, D. (2020): Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470.*

Asyaky, M. S. – Mandala, R. (2021): "Improving the Performance of HDBSCAN on Short Text Clustering by Using Word Embedding and UMAP," *2021 8th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, 1-6.

Blei, D. M. (2012): Probabilistic Topic Models. *Communications of the ACM,* 55(4): 77-84.

Blei, D. M. – Lafferty, J. D. (2006): Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*. New York: Association for Computing Machinery, 113–120.

Blei, D. M. – Ng, A. Y. – Jordan M. I. (2003): Latent Dirichlet Allocation. *Journal of Machine Learning Research,* 3: 993-1022.

Bouma, G. (2009): Normalized (pointwise) mutual information in collocation extraction. *Proceedings of the Biennal GSCL Conference*, 30: 31-40.

Bromley, J. – Guyon, I. – LeCun, Y. – Säckinger, E. – Shah, R. (1993): Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6: 737-744.

Devlin, J. – Chang, M.-W. – Lee, K. – Toutanova, K. (2018): Bert: Pretraining of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Egger, R. – Yu J. (2022): A Topic Modeling Comparison Between LDA, NMF, Top2Vec, and BERTopic to Demystify Twitter Posts. *Frontiers in sociology*, 7(886498): 1-16.

Grootendorst, M. (2022): BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794*., 1-10.

Hirschberg, J. – Manning, C. D. (2015): Advances in natural language processing. *Science,* 349(6245): 261-266.

Jánossy L. – Tasnádi P. (2016).: *Vektorszámítás II. – Vektorok és tenzorok differenciálása.* Akadémiai Kiadó. https://doi.org/10.1556/9789630598460.

Kaur, J. – Buttar, P. K. (2018): A systematic review on stopword removal algorithms. *International Journal on Future Revolution in Computer Science & Communication Engineering*, 4(4): 207-210.

McInnes, L. – Healy, J. – Astels, S. (2017): hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11): 205.

McInnes, L. – Healy, J. – Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426.*

Mifrah, S. – Benlahmar, E. H. (2020): Topic modeling coherence: A comparative study between LDA and NMF models using COVID'19 corpus. *International Journal of Advanced Trends in Computer Science and Engineering*, 5756-5761.

Muller, J. C. (1982): Non-Euclidean geographic spaces: mapping functional distance. *Geographical Analysis,* 14(3): 189–203.

Németh R. – Katona E. R. – Kmetty Z. (2020): Az automatizált szövegelemzés perspektívája a társadalomtudományokban. *Szociológiai Szemle,* 30(1): 44-62.

Reimers, N. – Gurevych, I. (2019): Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 3982–3992.

Röder, M. – Both, A. – Hinneburg, A. (2015): Exploring the space of topic coherence measures. *In Proceedings of the eighth ACM international conference on Web search and data mining*, 399-408.

Schofield, A. – Magnusson, M. – Mimno, D. (2017): Pulling out the stops: Rethinking stopword removal for topic models. *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. 2: 432-436.

Sia, S. – Dalmia, A. – Mielke, S. J. (2020): Tired of topic models? Clusters of pretrained word embeddings make for fast and good topics too!. *arXiv preprint arXiv:2004.14914*

Sievert, C. – Shirley, K. (2014): LDAvis: A method for visualizing and interpreting topics. *In Proceedings of the workshop on interactive language learning, visualization, and interfaces*, 63-70.

Silva, C. – Ribeiro, B. (2003): The importance of stop word removal on recall values in text categorization. *In Proceedings of the International Joint Conference on Neural Networks*, 3: 1661-1666.

Teh, Y. W. – Jordan M. I. – Beal, M. J. – Blei, D. M. (2004). Sharing clusters among related groups: Hierarchical Dirichlet processes*. Advances in neural information processing systems*, 17.

Teh, Y. W. – Jordan M. I. – Beal, M. J. – Blei, D. M. (2006): Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476): 1566-1581.

Temesi J. – Varró Z. (2017): *Operációkutatás*. Akadémiai Kiadó. https://doi.org/10.1556/9789630598699.

Terragni, S. – Fersini, E. – Messina, E. (2021): Word embedding-based topic similarity measures. *In International Conference on Applications of Natural Language to Information Systems*, 33-45.

Wallach, H. M. (2006): Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*. *New York: Association for Computing Machinery*, 977-984.

FÜGGELÉK

A. FELMERÜLT TECHNIKAI AKADÁLYOK ÉS MEGOLDÁSAIK

A szükséges csomagok telepítése, valamint a modell használata során felmerült hibaüzenetek és megoldásuk ismertetése hasznos lehet azok számára, akik alkalmazni szeretnék a bemutatott topikmodellek Python implementációját, így a kutatásom során ezeket részletesen dokumentáltam.

A *gensim* installálása először sikertelen volt, azt a hibaüzenetet kaptam, hogy *„ERROR: Could not build wheels for gensim, which is required to install pyproject.toml-based projects”*. A problémát az oldotta meg, hogy telepítettem az Anaconda-t és azon belül már sikeresen tudtam telepíteni a csomagot a *pip install gensim* paranccsal, mert az Anacondaban már alapértelmezetten telepítve vannak a legnehezebben installálható függőségei a csomagnak.

A BERTopic-hoz szükséges csomag telepítését Anaconda promptban a *pip install bertopic* paranccsal végeztem. Az első hibaüzenet arra vonatkozott, hogy nem sikerült a *hdbscan* alcsomag telepítése (*„ERROR: Could not build wheels for hdbscan, which is required to install pyproject.toml-based projects”*). A problémát a *conda install -c conda-forge hdbscan* parancs oldotta meg. A *conda* a *pip*hez hasonlóan egy csomagkezelő, amit az Anaconda fejlesztett ki. A *-c conda-forge* rész azt specifikálja, hogy a Conda-forge[37] gyűjteményből szeretnénk telepíteni a *hdbscan* csomagot. Ezután újra lefuttattam a *pip install bertopic*-ot, ami során a következő hiba merült fel: *„ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts. tables 3.8.0 requires blosc2~=2.0.0, but you have blosc2 2.5.1 which is incompatible”.* A probléma megoldása a blosc2 szükséges verziójának a telepítése volt: *pip install blosc2==2.0.0.* Ezek után Jupyter notebookban az *from bertopic import BERTopic* parancs továbbra is sikertelen volt a *„cannot import name 'is_nltk_available' from 'transformers.utils.import_utils'”* hiba miatt. Ezt a *from transformers.utils.import_utils import is_nltk_available* parancs lefuttatása a notebookban sem oldotta meg elsőre, viszont a Jupyter notebook bezárása és újraindítása után végre sikeres volt a *bertopic* importálása.

A modell első futtatása során a modellillesztő parancs (*topics, probs = topic_model.fit_transform(docs)*) a *„TypeError: sequence item 1: expected str instance, float*

[37] A Conda-forge egy közösség által létrehozott és vezetett gyűjteménye a condához szükséges csomagoknak. https://github.com/conda-forge (megnyitva: 2024.03.30.)

found" hibát generálta. Ennek az volt az oka, hogy a dokumentumok között volt üres cella is, ami a transzformáció során NaN-ná alakult, amit float típusú adatként kezel a Python. A probléma megoldása, hogy el kell távolítani az üres sorokat a dokumentumokat (beszédeket) tartalmazó oszlopból a listává alakítása előtt: *docs = data['speech'].dropna().tolist()*.

A *KeyBERTInspired()* első használatánál a *„AttributeError: 'NoneType' object has no attribute 'embed_documents'"* hiba fordult elő, mert ha *representation_model* paramétert is megadunk a *BERTopic()* modellnek, akkor muszáj *embedding_model*-t is beállítani, akkor is, ha előre definiált beágyazásokat használunk a teljesítményoptimalizálás miatt (ekkor alapesetben elhagyható ez a paraméter a *BERTopic()*-ból). Azért van erre szükség, mert a reprezentációk optimalizálásához a szövegbeágyazáshoz szükséges modellt is használja az algoritmus.

A *CountVectorizer*[38] funkcióban a korpusz-specifikus stopszavak kiszűrésére van egy *max_df* paraméter. Ebben a paraméterben lehet beállítani, hogy a szótárgenerálás során milyen dokumentum gyakoriság (a továbbiakban documentum frequency) fölött ignorálja a szavakat. Ha 0,0 és 1,0 közötti float típusú bemenetet (tizedes törtet) adunk meg, akkor a dokumentumok arányaként értelmezi a határt, ha egész számot adunk meg, akkor pedig darabszámként. Amikor ezzel a módszerrel próbáltam kiszűrni a túl gyakran előforduló szavakat, akkor az *„After pruning, no terms remain. Try a lower min_df or a higher max_df."* hibaüzenet érkezett. Ennek az volt az oka, hogy a *min_df* (tehát az a határ, aminél ritkábban előforduló szavakat nem vesz figyelembe) alapértelmezett beállítása 1, tehát egy darab dokumentum. A megoldás, hogy a *min_df*-et is meg kell határozni, és a *max_df*-nél kisebbre állítani. Én a *min_df*-et 0,0-ra állítottam (fontos, hogy tizedes törtként kell megadni a nullát is), hogy ne legyen alsó gyakorisági határ.

A *HDBSCAN()* paraméterezésénél a *min_cluster_size* beállításakor az *„IndexError: list index out of range"* hibát kaptam, mert túl magas számot adtam meg elsőre. A klaszterek mérete a korpuszunktól függ, így nincsen ennek a paraméternek egy alapértelmezett értéke, amt növelve vagy csökkentve befolyásolhatjuk a *BERTopic()* által megtalált topikszámot. A probléma megoldása, hogy csökkentsük a megadott számot. A paramétert változtatva tudjuk letesztelni, hogy a különböző minimális klasztérméretekre hogyan reagál a modellünk.

[38] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html (megnyitva: 2024.03.27.)