Szövegelemzési módszerek automatizációja

Szakács Béla Benedek, BME VIK Mérnökinformatikus BSc
Konzulens: Dr. Mészáros Tamás

Tartalom

1 Absztrakt

Napjainkban a természetes nyelvű szövegek elemzése a reneszánszát éli. Sokféle területen nélkülözhetetlenek, az ember-gép kommunikációtól kezdve a különböző adatbányászati alkalmazásokon át a szerzőiség-megállapításig. A módszerek igen széleskörűek, tisztán statisztikai, távolságvektorokon alapuló módszerektől kezdve SVM-eken (Support Vector Machine) át egészen klasszikus osztályozási feladatokat megoldó neurális hálózatokig sokféle eszközt felhasználnak. Ezen módszerek közül az én kutatásom a klasszikus, szövegelemekből képzett sokdimenziós koordináták közti távolságokon alapuló módszerre fókuszál.

Az általam vizsgált módszer elsődleges felhasználási területe a szerzőiség-megállapítás, de történtek egyéb feladatok megoldására történő kísérletek is, például a szerző korának vagy nemének megállapítása. A módszer egyik fontos problémája, hogy bár igen pontosan megállapítható a segítségükkel egy-egy mű kategóriája, a megfelelő módszer és a megfelelő paraméterezés megtalálása komoly kihívás, főként egy, statisztikai módszerekben és informatikában kevésbé jártas felhasználónak. Ennek a problémának a megoldására tett kísérlet alkotja a dolgozat gerincét. A kiindulási alapot egy már létező elemző, a stylo nevezetű R nyelvű csomag és az arra épülő, Shiny alapú webes kiegészítés jelentette.

Kidolgoztam egy olyan módszert, amely a szövegek előzetes elemzésére támaszkodva beállítja az elemzési paraméterek kiindulási értékét. Ez a módszer a vizsgált elemek gyakoriságát és a szövegek hosszát veszi elsődlegesen figyelembe, és az alapját ismert jellegzetességek képezik. A kezdeti paraméterértékek pontosabb beállítása érdekében kialakítottam egy eljárást, amelynek alapja a lokális keresés. Amennyiben rendelkezünk ismert kategóriájú szövegekkel, egy olyan paraméterezés, amely ezeket helyesen különíti el, valószínűleg helyesen fog ismeretlen kategóriájú szövegeket is elhelyezni ezekhez a már ismert kategóriájú szövegekhez képest. Ennek a paraméterezésnek a megtalálására a szimulált lehűlés lokális kereső algoritmusát használtam.

A kidolgozott módszerek gyakorlati alkalmazásához készítettem egy olyan webrendszert, amely felhőalapú, és egy vékonykliens csatlakozik hozzá a böngészőn keresztül. Ez nem igényel egyéni telepítést és konfigurálást, könnyen és intuitívan használható. Az eszköz webes felhasználói felülete tartalmaz varázsló és súgót a módszerben kevésbé járatos felhasználók segítésére, és nagyban leegyszerűsíti a használatot.

Ezt követően az elemző funkció hatékonyságát összehasonlítottuk más, manuálisan beállított paraméterekkel végzett kísérlet eredményeivel. Összehasonlítottuk laikus felhasználó által alkalmazott beállításokkal is, hogy értékelhessük, mennyire érte el a célját.

A paraméterek heurisztikus beállítása vagy kereséssel történő finomhangolása és a vékonykliens alapú architektúra jelentősen megkönnyíti a felhasználók számára az eredményes kísérletek elvégzését összetett elemzési módszerekre támaszkodva, különösebb előkészületek nélkül is.

2 Előszó

Az elmúlt években a természetes nyelvű szövegek elemzése a reneszánszát éli. Ahogy a mesterséges intelligencia több más területének, úgy ennek a fellendülésének is elsősorban a hirtelen nagymértékben megnövekedett számítási teljesítmény rendelkezésre állása az oka. A felhő alapú számítások elterjedése és az ugrásszerűen megnövekedett számítási kapacitás egyben azt is jelentette, hogy a szövegelemzés iránti igény a nagy, komoly szerverparkokkal rendelkező cégek privilégiumából egy, a mindennapokban is használható, elérhető termékké alakult át.

A téma tehát mindenképpen aktuálisnak mondható. A szövegelemzés iránti kereslet napjainkban elképesztően nagy. Szinte mindenütt, ahol nagy mennyiségű, természetes nyelvű szöveg formájában beérkező adattal kell megbirkóznia a rendszernek, előkerül a kérdés: hogyan? Ennek a kérdésnek a megválaszolása ugyan meghaladja ennek a dolgozatnak a kereteit, de a probléma egy kisebb szeletének elemzésével és megoldásával azért megpróbálkozom.

Az egyik kulcsterület, amivel részletesen foglalkozom, a szövegek különböző szövegjellemzők alapján történő osztályozása. Ezt a tudományágat összefoglalóan stilometriának nevezik. Erre létezik egy ismert és jól bevált módszercsalád, amely azonban sok paraméterrel rendelkezik, és ezek beállítása egy összetett, nemtriviális feladat. Ennek a problémának a megoldása alkotja a dolgozat gerincét. Az alapvető célkitűzésem az volt, hogy nagymértékben megkönnyítsem az informatikában nem jártas felhasználók számára a kísérletek elvégzését, és ehhez a paraméterek automatikus beállításával, és egy könnyen használható, minimális előkészítést igénylő felülettel járulok hozzá.

A dolgozatnak tehát kettős célja van: egyrészt igyekszik betekintést adni a stilometriába, a szövegelemzés egy speciális szakterületébe, másrészt a stilometria egyik fontos problémájára adott megoldást mutatja be.

3 Stilometriai bevezető

3.1 A stilometria fogalma

Maga a stilometria egy nehezen meghatározható tudományterület. Gyakorlatilag magában foglal mindenfajta stíluson alapuló szöveg-kategorizálási módszert, ezen felül pedig azok alalkalmazási módszereit és kísérletekben való megfelelő felhasználását.

Maga a tudományág ennélfogva három tudományterület határán fekszik: a statisztika, a statisztika és a nyelvészet (melyhez időnként csatlakozik az irodalomtudomány is) mind-mind fontos részét adják a kutatásoknak. Ez adja az egyik fontos problémáját a kutatásoknak, melyre később rátérek: a felhasználók problémáját.

A stilometria dióhéjban a stílus, mint a szöveg valamely aspektusától (szerzőjétől, témájától vagy típusától) függő tulajdonság számszerűsítése, szövegekből való kinyerése és különböző feladatokra való felhasználása. Természetesen kiterjeszthető nem csupán szövegekre, de zenére vagy hangfelvételekre is, illetve akármilyen lineáris, stílusbeli sajátosságokat tartalmazó adathalmazra.

A stílus, mint olyan, még az irodalomtudományban is egy neheze megfogható fogalom, és ez igaz a statisztikai és információelméleti megközelítésekre is. Egyszerűen nincs egy átfogó, mindenre kiterjedő modell, amely nyelvészetileg teljeskörűen leírja azt, hogy a stíluselemek pontosan mik, és mely szavak/szókapcsolatok hordozzák magukban.

Ennek megfelelően a stilometriai elemzések „stílus-alapú” megközelítése nem teljesen korrekt, de ugyanakkor bizonyos értelemben mégis pontosan erről van szó: az egész tudományág lényege, hogy megpróbálja ezt a bizonyos megfoghatatlan, nehezen definiálható fogalmat statisztikai adatokká redukálni. Pont emiatt a stilometria egy rendkívül sokszínű tudomány: az alkalmazott módszerek a statisztikai elemzés és a mesterséges intelligencia változatos területeiről érkeznek, és hatékonyságuk elemzésről elemzésre más és más.

Pont ezen okból nehéz a stilometriáról mint egzakt tudományról beszélni: ugyan valódi, megismételhető kísérletek viszik előre, a szabályszerűségek felderítése azonban még mindig gyerekcipőben jár, és kevés esetben alakult ki konszenzus módszerek általános felhasználhatóságát illetően. Még olyan, relatíve egyszerű kérdésekben, mint a szövegek szükséges minimális hossza egyes elemzési módszerekhez, sincsen teljes egyetértés a terület szakértői közt, és évről évre jelennek meg új, az eddigi eredmények átértékelésére késztető publikációk.

A stilometria tehát egy meglehetősen széles, egyelőre nagyrészt felfedezetlen tudományterület. Ugyanakkor folyamatosan jelennek meg publikációk a témában, és a kutatómunka egyik fontos része, a minél szélesebb körű felhasználás lehetővé tétele egyre nagyobb hangsúlyt kap. Online felületek, kész eszközként használható könyvtárak, szoftverek jelennek meg. Az ebben a dolgozatban leírt szoftver egyik legfontosabb, ha nem a fő feladata is pontosan a felhasználhatóság széleskörűbbé tétele.

3.2 Történeti áttekintés

Maga a stilometria nem újkeletű tudomány. Az első kísérletek még egészen a 19. század végéig nyúlnak vissza, Mendelhall 1887-es tanulmánya [1] Shakespeare színdarabjain, ezt követően pedig a 20. század első felében Yule (1934 [2], 1944 [3]) és Zipf (1932) [4] írt tanulmányokat. A stilometria máig gyakran idézett, példaértékű, talán a legnagyobb hatású kísérlete Mosteller és Wallace (1964) [5] nevéhez fűződik. Ez a kísérlet a *Federalist* nevezetű újság bizonyos, ismeretlen szerzőjű cikkeihez társított írókat, és egy Bayes-i statisztikai analízisen alapuló módszert használtak.

Ezt követően, főleg az 1990-es évektől kezdve a szerzőiség-megállapítási kísérletek leginkább a stílus, mint számszerűsíthető szövegtulajdonság vizsgálatán alapultak, ezt nevezzük stilometriának. Habár a 20. század közepén még nem álltak rendelkezésre a szükséges eszközök nagyméretű korpuszok hatékony, gyors elemzésére (sőt, maguknak a korpuszoknak a beszerzése és megfelelő formátumra hozása is problémát okozott), ez a 90-es években megváltozott.

Az internet elterjedésével, és a nagy teljesítményű számítógépek, valamint algoritmusok elterjedésével megnyílt az út a stilometria előtt. Az internetes kommunikáció és a rengeteg digitalizált szöveg felkerülése jelentősen egyszerűbbé tette nagyméretű, kevés utómunkát igénylő korpuszok beszerzését. A számítási kapacitás drasztikus növekedése lehetővé tette a rendkívül alapos, hosszú futási idejű elemzéseket. Megnyílt a lehetőség új területeken való felhasználásra, például törvényszéki felhasználás (Chaski, 2005 [7]; Grant, 2007[6]), számítógépes vírusok szerzőinek megállapítása (Frantzesku, Stamatatos, Gritzkalis, & Katsikas 2006 [8]), de irodalmi elemzéseket is folytattak vele (Burrows 2002 [9], Hoover 2004 [10]).

A kétezres évek első évtizedét a stilometria fellendülése jellemezte. Rengeteg módszert fejlesztettek ki, számtalan kísérletet végeztek, és összességében a tudományág elindult a fejlődés útján. Az elmúlt években is folyamatosan látnak napvilágot újabb módszerek, újabb kísérletek (példa: a Grimm testvérek műveinek analízise több különböző módszerrel (többek között kézírás-elemzéssel és stilometriai elemzéssel is.) [13]).

Ahogy a tudományág folyamatosan fejlődik, folyamatosan vannak kísérletek arra, hogy a módszereket, eszközöket az informatikában kevésbé jártas, kisebb erőforrásokkal rendelkező kutatók számára is elérhetővé tegyék [22]. Ez a dolgozat is egy ilyen kísérletet mutat be.

4 Szövegelemzési módszerek

A természetes nyelvű szövegek feldolgozásának egyik fő ága a kategorizáció. A stilometria, mint fentebb láttuk, pontosan ezt célozza meg. A stílus-kategorizációra több különböző módszer létezik, melyek közül a leggyakrabban használtakat alább ismertetem:

4.1 Statisztikai módszerek

Az egyik leggyakrabban használt és legfontosabb elemzési módszerkört a statisztikai módszerek jelentik a stilometriában. A későbbiekben részletesebben tárgyalt egyéb, a mesterséges intelligencia módszereit használó módszerek sem voltak képesek kiszorítani, ennek elsődleges oka az, hogy az esetek jelentős részében szigorúbb követelményeknek kell megfelelnie az adatoknak az ottani felhasználáshoz, mégsem biztosítanak szignifikánsan jobb eredményt.

Amikor statisztikai módszerekről beszélünk, egy, alapjaiban meglehetősen egyszerű módszerről van szó. Az eljárás a „bag of words" módszert használja (bár a „bag of features" elnevezés valamelyest jobb lenne, lásd a Vizsgált elemek fejezetet), és valójában nem foglalkozik a szavak sorrendjével. Az elemzésre kiválasztott szövegek mindegyikéből kigyűjti az előforduló szavakat, mindegyikhez hozzárendeli egy számot, amely a szövegben való előfordulásainak száma, majd az így kapott n-dimenziós koordinátavektort (n a különböző szavak száma) használja fel.

A feltételezés a következő: a stílus-lenyomat megtalálható a szavak gyakoriságában. Ennek megfelelően a fent említett koordináta-vektorok közti távolságok kiszámításával eljuthatunk a vizsgált szövegek halmazokba rendezéséhez, és ily módon következtetéseket tudunk leszűrni az ismeretlen szerzőjű szövegek ismert szerzőjű szövegekhez képest való elhelyezkedéséről vagy csupán a szövegekből kialakuló klaszterekből.

A módszert először Burrows írta le, és utána azóta is Burrows Deltájaként emlegetjük a leggyakrabban használt verzióját [9] (nem tévesztendő össze a távolságmértékkel, amelynek szintén ez a neve).

A módszer két fontos tervezési problémát vet fel: milyen egységeit használjuk fel a szövegnek az elemzéshez, és milyen mértéket használjunk arra, hogy n-dimenziós, gyakran különböző hosszúságú vektorok közti távolságot megállapítsuk? A következő két szakasz ezeket a kérdéseket tárgyalja.

4.1.1 Vizsgált elemek

Habár az előbb a „bag of words" módszerről volt szó, az elemzés során nem csupán szavakra lehet bontani a szöveget. Habár még mindig ez az alapértelmezett módszer, és ezt tükrözi több paraméter neve is az elemzés során, a kutatások hamar kimutatták, hogy betű n-gram-ok (n karakter hosszúságú betű-sorozatok) és szó n-gram-ok gyakran még jobb eredményre vezetnek. A fő problémát a betű n-gram-ok esetén az jelenti, hogy a kapott „szavak" jelentős része nagyon kevés alakalommal fog előfordulni a szövegben, és a távolságmértékeknél illetve egyéb paraméterek beállításánál feltétlenül figyelembe kell vennünk ezt. A szó n-gram-ok esetén ugyanez a probléma,

csak ez esetben még érzékenyebb a rendszer a nagyobb n-ekre, mivel 3 vagy annál nagyobb n-ek használata esetén rövid szövegeknél előfordulhat, hogy gyakorlatilag nem találunk 2-3-nál többször előforduló szövegelemeket. Ez pedig értelemszerűen nagyban csökkenti az elemzés hatékonyságát.

Ezen problémák mellett is vannak a szó és betű n-gram-oknak elvitathatatlan előnyei. Azon nyelvekben, ahol a ragozás összetett és rétegzett, a szavak használata nem mutatható ki csupán a szavak gyakorisága alapján (a magyar nyelven a képző-jel-rag alapú szóképzés miatt ez a probléma még erőteljesebben jelentkezik), de ez a módszer, amellett, hogy relatíve sok „fölösleges" adatot termel, kiválóan alkalmas a szótövek közti hasonlóságok megtalálására és a ragok okozta hibák kiküszöbölésére.

4.1.2 Távolságmértékek

Habár n-dimenziós koordináták közti távolságok kiszámítására rendkívül sokféle módszer létezik, az alábbiakban csak pár, gyakrabban használt módszert tárgyalunk. Ezek egy része kifejezetten stilometriai elemzésekre lett kifejlesztve, míg mások jól ismert matematikai módszerek. A képletekben az n a szólista hossza, A és B a szövegek, A_i és $B_i$pedig az i-edik szó gyakorisága a szövegekben.

Az *Euklideszi távolságmérték* a legismertebb ezen utóbbiak közül. Mindenki ismeri és alkalmazza. Sajnos, mivel a szavak eloszlása nem ugyanolyan, ezért a stilometrai elemzésekben ritkán kerül felhasználásra, elsősorban ritkább szavak esetén használhatóak hatékonyan.

$$\delta_{(A,B)} = \sqrt{\sum_{i=1}^{n} |(A_i)^2 - (B_i)^2|}$$

A Manhattan távolságmérték is egy jól ismert, gyakran használt módszer.

$$\delta_{(A,B)} = \sum_{i=1}^{n} |A_i - B_i|$$

Burrows (klasszikus) Deltája [9] nagy előrelépést jelentett az eddig használatos távolságmértékekhez képest, és mind a mai napig az egyik leggyakrabban használt távolságmérték. Gyakran használják benchmarknak más módszerekkel való összehasonlítás céljára. A μ_i és a σ_i a vizsgált szó átlagos egész korpusz-béli előfordulását és az egész korpusz-béli szórását jelöli.

$$\Delta_{(A,B)} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{A_i - \mu_i}{\sigma_i} - \frac{B_i - \mu_i}{\sigma_i} \right|$$

Argamon lineáris Deltája [23], avagy gyakrabban csak Argamon Deltája egy szintén gyakran használt mérték. Ahogy Burrows deltája, úgy ez is igen érzékeny a korpuszban található szövegek számára, azaz fontos, hogy egyazon csoportból nagyjából ugyanannyi szöveg álljon a rendelkezésre.

$$\Delta_{(A,B)} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\left|\frac{(A_i)^2 - (B_i)^2}{\sigma_i}\right|}$$

Eder Deltája [14] egy módosított verziója a Burrows Deltájának. A gyakoribb szavak súlyozását megnöveli valamelyest, a ritkábbakét pedig lecsökkenti. Az n_i a vizsgált i-edik szó helye a listában, felülről.

$$\Delta_{(A,B)} = \frac{1}{n}\sum_{i=1}^{n}\left(\left|\frac{A_i - B_i}{\sigma_i}\right| \times \frac{n - n_i + 1}{n}\right)$$

Eder egyszerű távolságmértéke [14] igazából egy rendkívül egyszerű, de meglepően hatékony módszer, amely különösen olyan esetekben bizonyult hatékonynak, ahol az összetetteb metrikák a körülmények miatt nem működtek olyan hatékonyan.

$$\Delta_{(A,B)} = \sum_{i=1}^{n}\left|\sqrt{A_i} - \sqrt{B_i}\right|$$

A Canberra távolságmérték [24] egy, bizonyos esetekben rendkívül jó eredményeket produkáló metrika, amely nagyon érzékeny a ritkább szavakra, de pont emiatt a zaj nagyon nagymértékben rontja a teljesítményét.

$$\delta_{(A,B)} = \sum_{i=1}^{n}\frac{|A_i - B_i|}{|A_i| + |B_i|}$$

A Koszinusz távolság is egy gyakran használt, jól ismert mértékegység. Habár nem túl gyakran használt, említésre méltó.

$$1 - \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2} \cdot \sqrt{\sum_{i=1}^{n} {B_i}^2}}$$

4.1.3 MFW

A rövidítés a Most Frequent Words szóhármast takarja, habár itt is pontosabb lenne szavak helyett szövegelemekről beszélni. Ez a paraméter is a legfontosabbak között van: egy százalékos érték, amely meghatározza, hogy a teljes szólista hány százalékát használjuk fel az elemzés során, kezdve a leggyakoribb szavaktól.

Az MFW mint paraméter azért bír kiemelt jelentőséggel, mert nagymértékben hatással lehet az elemzés kimenetelére, és mindemellett a megfelelő érték megtalálása is nehéz feladat. Nagymértékben függ a szövegek hosszának eloszlásától, de a szava eloszlása is komoly befolyásoló tényező. Ezen kívül még sok egyéb, nehezen előrelátható tulajdonsága van a korpusznak, amely hatással van az MFW optimális értékére.

Az MFW nem csupán statikus paraméterként jelenik meg. Több elemzési módszer használ változó MFW értéket, valamilyen relatíve alacsony kezdőértékkel és folyamatosan növelve egy fix

lépésközzel minden egyes iteráció során. Ez, pont a nehezen meghatározhatóság miatt, ellenállóbbá teszi az elemzést az ilyen típusú hibák ellen.

4.1.4 Culling

Ez a másik fontos százalékos paraméter, amely nagymértékben hatással van az elemzés minőségére és eredményére. A Culling egy feltételt szab a vektorok kialakításához használt szavak természetére: az itt megadott szám jelentése, hogy a szónak a korpuszban található szövegek hány százalékában kell előfordulnia ahhoz, hogy felhasználásra kerüljön az elemzés során.

Ez a paraméter, az MFW-hez hasonlóan nehezen megjósolható, habár a szövegek hossza és a szövegek szóhalmazai közöti átfedés elég jól meghatározza az optimális mértéket. A Culling ugyanakkor erősen függ a fentebb említett vizsgált elem típusától. Minél ritkább elemekkel dolgozunk, annál nagyobbra kell venni a Culling értékét. Könnyedén belefuthatunk ugyanis abba a problémába, hogy például szó 3-gramok esetében már egy egészen alacsony, 10-15 %-os Culling is azt eredményezi, hogy az elemzés során statisztikailag elhanyagolható hosszúságú vektorok maradnak, amelyek nem alkalmasak a további elemzésre.

Ahogy az MFW, úgy a Culling is százalékos érték. Ennek megfelelően természetes az az ötlet, hogy ezt is iteratív módon, folyamatosan változtatva használjuk az elemzés során, kiküszöbölve a pontos beállítás hiányából adódó gyengeségeket. Ez, bár egy alkalmazott módszer, mégsem olyan hatékony, mint az MFW esetén, mert akár egészen kis változásokra is nagyon érzékeny az elemzés.

4.1.5 Szólista levágása, szavak eltávolítása

Az elemzés során természetesen a fent említett négy paraméteren felül még meglehetősen sok egyéb paraméter is létezik, amelyeket az elemzések során használnak. A korpusz nyers formában ritkán alkalmas az elemzésre, minden nyelvben vannak olyan szavak (úgynevezett „stopword” -ök), amelyeket nem használnak fel elemzéseken, mivel kísérletek kimutatták, hogy ezen szavak nem javítják a teljesítményt. Ezek a szavak jobbára névelők vagy kötőszavak, olyan elemei a nyelvnek, amelyek használata elkerülhetetlen és a gyakoriságuk miatt nagymértékben hatással vannak az elemzés eredményére.

Ezen kívül a teljes, gyakoriság szerint rendezett szólistát felülről és alulról is le lehet vágni. Az alját (a nagyon ritka szavakat) akkor érdemes levágni, ha meglehetősen hosszú szövegeket elemezünk, és sok olyan szó van, amelyek ritkán fordulnak elő. Az MFW megfelelő beállításán felül még érdemes egy bizonyos ponton (pár ezer szónál) elvágni a listát, hogy ezzel megkönnyítsük az elemző dolgát számításigény szempontjából. Nagyon nagy vektorok esetén ugyanis a futási idő rendkívüli módon meg tud nőni.

A lista tetejének levágása akkor javasolt, ha valamilyen oknál fogva vannak nagyon gyakori szavak, amelyeket semmiképpen nem akarunk bevenni az elemzésbe. Ennek elsődleges oka lehet az, hogy a nyelv, amit vizsgálunk, tartalmaz olyan „stopword” -öket, amelyeket a szoftver nem ismer fel magától, és ez egy jó megkerülése a problémának, illetve, ha a szöveg típusa miatt vannak

olyan szavak, amely rendkívül gyakran fordulnak elő, pedig a nyelvben nem szoktak (pl. egy regényben a szereplők nevei, vagy az „ő”).

4.2 Vizualizációs módszerek

A végeredménye az elemzéseknek alapvetően egy táblázat, amely minden két szöveg között megmutatja a távolságot. E mögött természetesen rengeteg egyéb adat is van, de ez a táblázat jó kiindulási alapot jelent a stílus-kategorizációs kísérletekhez, és főleg alkalmas egy-egy módszer kiértékelésére, mivel minden távolság meg van adva, tehát egyszerű valamilyen algoritmussal kimutatni, hogy az egyes szerzőkhöz tartozó művek mennyire közel kerültek egymáshoz, és erre valamilyen mértéket adni.

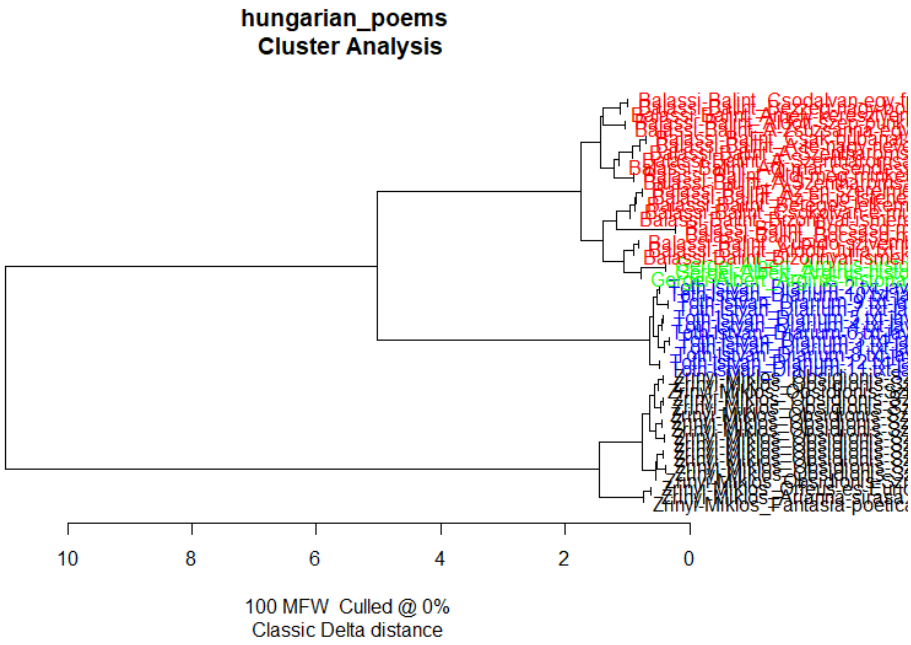
Ez a táblázat azonban, főleg sok szöveg esetén nagyon semmitmondó egy emberi szemlélő számára. Emiatt szükség van valamilyen, az emberi szemlélők számára könnyebben és intuitívabban értelmezhető módszerre. Szerencsére a statisztika tudománya több ilyet is ismer. Elsősorban olyan módszereket keresünk, amelyek alkalmasak sok pont megjelenítésére úgy, hogy az egyes pontok közti távolságok jól látszanak, és az egy halomba kerülő szövegek felismerhetők.

Természetesen az alább bemutatott lehetőségek nem fedik le a teljes tárházat, de ezeket gyakran használják és a tapasztalatok szerint jól alkalmazhatóak csoportok megtalálására és stílus-kategorizációs kísérletekben. Alább három ilyet mutatok be: a klasztereket, a PCA-t és a konszenzusfát.

4.2.1 Klaszterek

A klaszterek használata egy régóta ismert, jól bevált technika. Alapvetően a dendogramot használják a megjelenítésre: ez egy jól ismert, bevált módszer sok elem egymáshoz képesti távolságának megjelenítésére. Az egyes leágazások mutatják, hogy az ezt követően különböző ágakon található elemek ekkora távolságra vannak egymástól. Az alján látható skála mutatja meg, hogy pontosan mekkora távolságánál van az elágazás.

A dendogram és összességében a klaszterezés akkor használatos, ha kevés MFW-t vizsgálunk meg. Sok MFW-hez alkalmasabb a későbbiekben tárgyalt konszenzusfa. A felhasználó számára a dendogramról azonnal leolvasható, hogy mely szövegek vannak közelebb egymáshoz, és mely szövegek vannak nagyon távol. Amennyiben az ábrán szépen kirajzolódnak az egyes szerzőkhöz tartozó csoportok, a paraméterezés megfelelő.

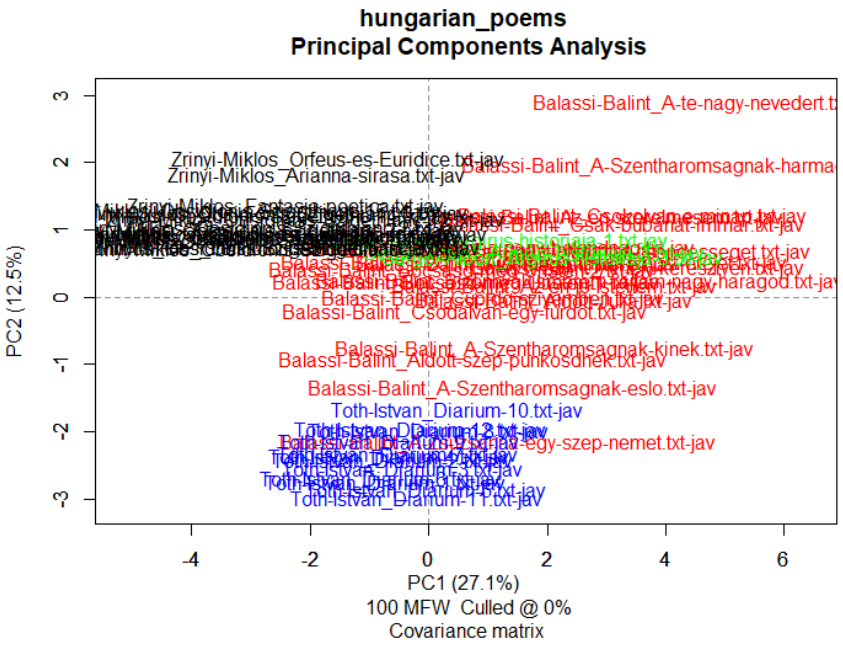


1. ábra: dendogram - a színek reprezentálják a különböző szerzőket

4.2.2 PCA

A Principal Component Analysis rövidítése, magyarul a legfontosabb komponens alapán történő elemzés. A lényege, hogy a két legfontosabb tulajdonság szerint egy síkon ábrázoljuk pontok halmazaként a szövegeket. Ez az elemzési módszer kiválóan tudja szemléltetni a szövegek eloszlását, ha jól választjuk ki a komponenseket.

A módszer fő hátránya, hogy rendkívül ritkán szembesülünk olyan helyzettel, ahol csak két szó gyakorisága alapján történő PCA is képes összefoglalni az elemzés eredményét. Sajnos, pont amiatt, hogy az esetek többségében több száz szóból álló vektorokkal dolgozunk, a PCA önmagában meglehetősen haszontalan. Több szókombinációra használva azonban fontos következtetésekre juthatunk általa, és jól kiegészít más, a távolságokat jobban megjelenítő, de az egyes szavak szerinti statisztikákat gondosan elfedő módszereket.

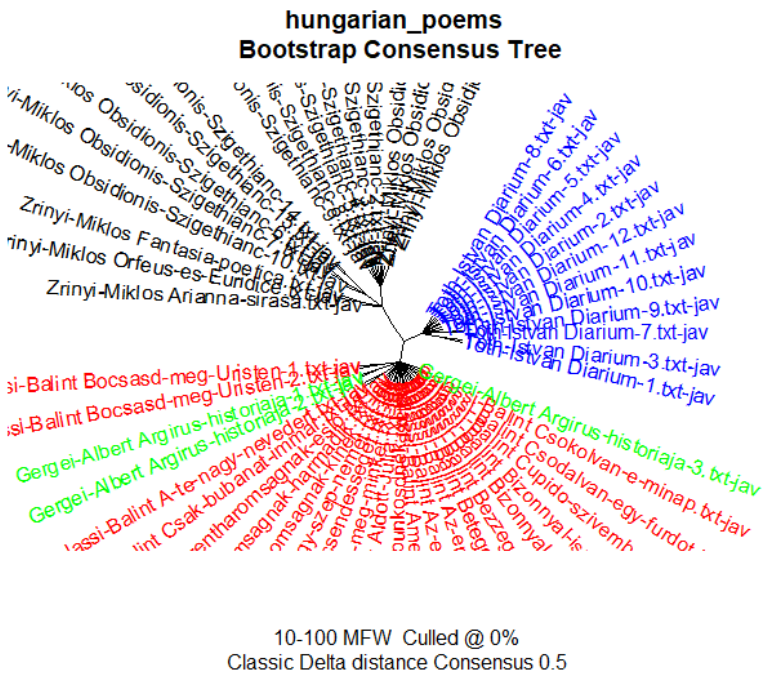


2. ábra: PCA - a színek reprezentálják a különböző szerzőket

4.2.3 Konszenzusfa

Ez a módszer főleg akkor használatos, amikor több MFW értéket kívánunk vizsgálni, és ezek összességét venni az elemzés végén. A konszenzusfa gyakorlatilag egy sorozatnyi klaszterezés eredményeit összegzi egy ábrába. Valamelyest kevésbé látszanak rajta a távolságok, elsősorban arra való, hogy jól megfigyelhetőek legyenek az egyes csoportok, illetve kiválóan alkalmas azon szövegek megtalálására, amelyek nagyon nem illenek a többi közé. Ezek a konszenzusfán általában messze a többitől, kilógva helyezkednek el.

A konszenzusfa másik fontos különlegessége, hogy egy plusz paramétert is magával hoz: a konszenzus erősségét. Ezzel a paraméterrel lehet megadni számára, hogy a klaszterezési eljárások hány százalékában kellett közvetlen kapcsolatot találnia a két szöveg között ahhoz, hogy itt is közvetlen kapcsolatban levő szövegként jelenjenek meg.



3. ábra: Konszenzusfa – a színek reprezentálják a különböző szerzőket

4.3 SVM, k-NN, Naiv Bayes és NSC

Ezen módszerek, habár a dolgozat szempontjából kevésbé fontosak, mindenképpen említést érdemelnek, mint egzotikusabb, a mesterséges intelligencia területén belül más problémákra gyakran használt megoldások.

Az SVM (Support Vector Machine) a mesterséges intelligencia egy jól bevált eszköze. Ezek ellenőrzött tanulásos modellek, amelyek kiválóan alkalmasak kategorizációs problémák megoldására. Ennek megfelelően ideális jelöltek a szerzőiség-megállapítási vizsgálatok elvégzésére. Mindemellett már meglehetősen régóta ismertek, az eredeti algoritmus 1963-ban született meg Vladimir N. Vapnik és Alexei Ya. Chervonenkis tollából.

Az SVM a stilometriában rendkívül népszerű. Gyakorlatilag a klasszikus statisztikai módszerek mellett ez a leggyakrabban használt algoritmus. A legtöbbször elegendő lineáris SVM-eket alkalmazni (habár vannak polinomiális és egyéb módszerek is), amelyek az adathalmazt lineáris hipersíkkal szeparálják el az n-dimenziós térben. Az algoritmus ennek a megtalálására szolgál egy tanító adathalmaz segítségével, majd egy ellenőrző adathalmaz használatával megvizsgálható a tanulás hatékonysága.

A k-NN módszer (k-nearest neighbours) egy másik gyakran alkalmazott algoritmus kategorizációs problémák megoldására. Gyakorlatilag a k db legközelebbi szöveget vesszük minden szöveg esetén, és megvizsgáljuk, hogy ezek közül melyik íróhoz tartozik a legtöbb. A szöveg ehhez az íróhoz fog tartozni az algoritmus szerint. Ez egy jó módszer a fentebb említett távolságmértékek felhasználására stílus-kategorizációhoz.

A k-NN módszer nem annyira független, önálló módszerként van jelen a stilometriában, inkább a már létező távolságszámítási módszerek kiegészítésére. A k paraméter megtalálása ugyanakkor általában nemtriviális probléma, és mindemellett vannak finomabb, a távolságokat alaposabban figyelembe vevő algoritmusok erre a feladatra.

A naiv Bayes-osztályozás egy kiváló valószínűségalapú osztályozó algoritmus, mindössze egyetlen problémája van: feltételezi, hogy a vizsgált paraméterek függetlenek egymástól, ez pedig a stilometria esetén sajnos ritkán egyértelmű. Pont emiatt nem használatos olyan gyakran elemzésekben, mint a fentebb említett SVM-ek. Ugyan futási idő tekintetében sokkal jobb náluk, a teljesítménye nem kárpótol eléggé.

Az NSC (Nearest Shrunken Centroid) is egy osztályozási modell. A lényege, hogy a minden szöveget megvizsgálunk a következő módszerrel: az ismert, ugyanazon szerzőhöz tartozó szövegek csoportjaihoz hozzárendelünk egy átlagot, majd ezeket az átlagokat egy előre meghatározott mértékben az összes átlag átlaga felé mozdítjuk el. Ezek után megvizsgáljuk, hogy a kiválasztott szöveg melyik csoport átlagához van legközelebb, és ahhoz a szerzőhöz kapcsoljuk.

Az NSC előnye, hogy szövegek osztályozására egy jól bevált, kipróbált algoritmust ad. A hátránya, hogy bár bináris esetekben (két csoportnál) igen jó eredményeket ad, több csoportnál a hatékonysága jelentősen lecsökken.

5 Stylo és egyéb módszerek elemzése

5.1 A stylo csomag

Mivel a dolgozatban bemutatott megoldás alapját a stylo könyvtár jelenti, ezért ennek bemutatása elengedhetetlen.

A stylo csomag egy R-ben írt függvénykönyvtár, amely a fentebb említett elemzési módszereket és még sok egyéb hasznos, stilometriai elemzések során gyakran használatos funkciót tartalmaz. A csomag nyílt forráskódú, GitHub-on megtalálható az egész projekt. A készítői Maciej Eder, Jan Rybicki és Mike Kestemont.

A stylo csomag elsődleges funkciója, hogy egyszerűen használható és kényelmes legyen. Ennek megfelelően a csomag, habár rendkívül sok függvényt tartalmaz, az elemzések elvégzéséhez inkább kevés, de rendkívül részletesen paraméterezhető függvény áll a rendelkezésre. Mivel a stylo-hoz a korpusz előkészítése meglehetősen egyszerű (egy corpus elnevezésű mappában kell elhelyezni a fájlokat az aktuális workspace-en belül az R session-ben), ezért alább az elemzést végző függvényekre fókuszálok.

5.1.1 Stylo

Ez a csomag névadó függvénye, és mint ilyen, a leggyakrabban használt. Rendelkezik egy meglehetősen minimalista GUI-val is, de a paraméterezés R-ből is könnyen használható. A stylo egy rendkívül összetett függvény: ugyan csak szimpla statisztikai elemzések elvégzésére képes, mégis rengeteg paramétert lehet rajta beállítani az elemzéshez. A Statisztikai módszerek fejezetben említett összes lehetőségen felül rengeteg, a grafikus megjelenítés finomhangolására szolgáló beállítást lehet rajta elvégezni. A stylo függvény GUI-ja öt csoportra osztja a beállításokat:

Az *input & language* csoportba tartozik a korpusz különböző tulajdonságainak megadása. Itt lehet beállítani a használt szövegek formátumát (pl. xml, html), illetve a szövegek nyelvét, amely talán a legfontosabb beállítás ezek közül, hiszen ettől függ több további beállítás is. Ezen felül a stylo alapvetően ASCII kódolásban dolgoz fel szövegeket, de meg lehet adni itt, hogy UTF-8-as kódolású szövegeket dolgozzon fel.

A *Features* csoport tartalmazza az elemzés legfontosabb beállításainak jelentős részét. Itt kell megadni azt, hogy szó vagy karakter n-gram-okat használunk az elemzés során, és az n-et. Az MFW és a Culling is itt kerül beállításra, mint ahogy a szólista elejének és végének levágása is. A többi, itt elhelyezett beállítás a korpusz korábbi elemzéseiből megmaradt adatok felhasználásához nyújt segítséget.

A *Statistics* csoport foglalja magába az analízis végeredményének megjelenítését, illetve a rendkívül fontos távolságmérték-beállítást.

A *Sampling* csoport tartalmazza a szövegből való mintavételezési módszerrel kapcsolatos beállításokat. Amennyiben a szövegek hossza között nagyságrendi eltérések vannak, mindenképpen szükséges mintavételezést használni.

Az Output csoport pedig már csak a megjelenítéssel kapcsolatos beállításokat tartalmaz, hogy a kimenet kinézete minél pontosabban testreszabható legyen, illetve az elkészült grafikonok lementésével kapcsolatos beállítások (formátum, méret stb.)

5.1.2 Classify

Ez az a függvény, amelybe be lettek építve a modern, mesterséges intelligencia alapú megoldások (SVM, k-NN, Naiv Bayes és NSC fejezet). Ezen felül ez az a függvény, amely alapvetően a szövegek szerzőkhöz rendeléséhez lett tervezve, és örököli a stylo-ban implementált megoldásokat is. Az alábbiakban csak az extra beállításokkal foglalkozunk:

A Statistics csoport kiegészült a Delta, k-NN, SVM, NaiveBayes és NSC beállításokkal, amelyekkel fentebb már foglalkoztunk. A Delta a klasszikus statisztikai módszereket takarja, mivel a módszer egyik első verziója Burrows Delta nevezetű algoritmusa volt [9].

Ezen felül a felület tartalmaz még ezekhez a módszerekhez kapcsolódó további beállításokat, úgymint a k-NN és az NSC paramétereinek értékét.

5.1.3 Rolling.delta, Rolling.classify

Ez a két függvény arra a feladatra került be a csomagba, hogy ha egyetlen szövegen belül kívánunk stílusbeli változásokat, eltéréseket. Ablakos módszert használ, azaz a szöveget felosztja egyenlő méretű ablakokra (ez kézzel paraméterezhető), majd minden ablakra elvégzi az elemzést, és a végén kijelzi a talált stilometriai változásokat a szövegben.

5.1.4 Oppose

Végrehajt egy kontrasztív analízist két szövegcsoport között. Több, a stilometriában használatos módszer áll a rendelkezésre ehhez, de ezzel a résszel nem fogok foglalkozni a továbbiakban, mivel a dolgozat keretein kívül esik ezen módszerek részletesebb tárgyalása.

5.2 A Shtylo

A stylo elsődleges problémája, hogy mindenféleképpen szükséges hozzá némi informatikai tudás. Ismerni kell az R nyelvet legalábbis alapszinten, főleg a bonyolultabb kísérletek elvégzéséhez. A stylo, bár rendelkezik GUI-val, nem tartalmaz semmiféle magyarázatot hozzájuk. Rendelkezésre áll a csomaghoz egy pdf dokumentum [14], amely részletesen tárgyalja a funkciókat, de annak értelmezéséhez is szükséges a stilometriai módszerek ismerete és alapvető statisztikai tudás.

A stylo önmagában tehát nem igazán alkalmas széleskörű felhasználásra. Irodalomkutatók, nyelvészek és egyéb, az informatika terén kevésbé jártas felhasználók számára a stylo egyszerűen nagyon nehezen használható.

Ennek a problémának a megoldására született meg a Shtylo, a stylo Shiny-ban írt kiterjesztése. A rendszert Dobi Jan Sándor fejlesztette ki. A szoftver elsődleges célja a számításigényes műveletek szerveroldalra való delegációja volt: ily módon a felhasználó egy szimpla vékonykliensen, akár egy webböngészőn keresztül is képes a kísérletek elvégzését irányítani, míg

maga a statisztikai számítás a szerveren fut, akár felhőalapú megoldásokon is, amennyiben szükséges. A program igényel egy MongoDB alapú szervert, amelyet a korpuszok tárolására szolgál, ez egyébként független magától a Shiny backendtől.

A szoftver sok problémára megoldást ad: végre használható otthonról, megfelelő környezet telepítése és egyebek nélkül is. Mindazonáltal a Shtylo továbbra is csupán egy egyszerű kivezetése a stylo függvénynek, nem is implementálja a többi függvényt, és mindössze minimális támogatást nyújt korpusz-menedzsmentre. Ezek mellett pedig aki nem ismeri a stylo dokumentációját vagy nem járatos eléggé a stilometriában, hogy az alapján el tudja sajátítani a használatát, az még mindig nem lesz képes a szoftver használatára.

Ennek a problémának természetesen több megoldása is létezik. Dobi Jan Sándor részletesen leírja a Shtylo telepítését és használatát, ennek elolvasása után egy laikus felhasználó számára sem jelent gondot a szoftver használata, ugyanakkor a stilometriai összefüggések megértése nélkül az elemzések jó eséllyel kudarcra vannak ítélve.

A dolgozat témájául ennek a problémának az összetett megoldási kísérlete szolgál, amely a Shtylo kibővítésén alapul. Alább bemutatásra kerül a formai megoldás, amely egy varázsló és egy súgó képében került elkészítésre, és igyekszik hasznos tanácsokkal szolgálni a felhasználó számára a szoftveren keresztül. Ez a módszer természetesen a legegyszerűbben megvalósítható és legkevesebb problémát jelentő funkció.

Ezen felül természetesen elkészült egy összetettebb rendszer is a háttérben: egy automatikus paraméterező, amely megkísérli a kézi paraméter-beállítást kiváltani. Ebből két verzió is készült, egy egyszerűbb, heurisztikákon alapuló, és egy összetettebb, a stilometriában gyakran használt előzetes paraméterhangolást automatizáló, lokális keresést használó megoldás.

5.3 A varázsló és a súgó

A fentebb tárgyalt problémára adott első megoldási kísérlet tehát egy egyszerű súgó és varázsló elkészítése volt. Ezek elkészítése során az egyetlen kihívást a szöveg megfelelő megfogalmazása jelentette: a súgóban megjelenő szövegeknek laikus felhasználók számára is könnyen érthetőnek kell lennie.

A varázsló funkciója a következő: igyekszik egy intuitív, könnyen használható, kézreálló felületet adni a felhasználó számára, ahol megjelenhetnek a súgóhoz tartozó elemek és a felhasználó könnyen és egyszerűbben képes kiválasztani azt a módszert, az elemzésre a legalkalmasabbnak tűnik.

A súgó két rétegben helyezkedik el a varázsló felületén: egyrészről amennyiben a felhasználó az egerét valamelyik bemeneti mező fölé mozgatja, egy felugró tooltip megjelenít alapvető információkat az adott mezőről. A másik réteg a varázsló minden oldalán megjelenő oldalsó szöveges mező, amely általános áttekintést és tanácsokat tartalmaz az adott oldalon található mezőkkel kapcsolatban.

A varázsló tehát gyakorlatilag átveszi a Shtylo beállítási felületét, de becsukható mezők helyett oldalakon jeleníti meg a beállításokat, hogy kevésbé terhelje le vele a felhasználót. Az egyes oldalak közti navigációért az oldal alján található gombok felelősek, de az oldal tetején található sáv segítségével a felhasználó szabadon is navigálhat az oldalak között.

A varázsló felülete rendelkezik még egy fejlesztéssel a Shtylo-hoz képest: a beállított paraméter-kombinációt a felhasználó a felület alján található mező segítségével el tudja menteni egy tetszőleges elnevezésű fájlba, és később be tudja onnan olvasni azt. Ez nagy könnyebbséget jelent több kísérlet elvégzése esetén, illetve a kísérlet reprodukcióját is könnyebbé teszi.

6 Saját munka

6.1 Célkitűzések

A szoftverhez tehát elkészült egy súgó és egy varázsló felület, hogy megkönnyítse a felhasználók munkáját. Ez azonban még nem oldja meg a fő kérdést: a laikus felhasználó még mindig nem képes pontos paraméterezést összeállítani. Ennek megfelelően szükség volt egy jobb, pontosabb módszerre, amely képes elvégezni a paraméterek beállítását.

A legfontosabb cél a következő volt: kiválasztani azokat a paramétereket, amelyek szignifikánsan befolyásolják az elemzés eredményét, és aztán ezekre egy algoritmust konstruálni, amely képes ezek értékét optimalizálni.

Ennek megvalósítására két utat választottam: egy heurisztikus elemzőt, amely kis számítási teljesítményt igényel, és egy lokális kereséses algoritmust, amely azonban nagy számításigényű és jelentősen lassabb, cserébe sokkal megbízhatóbb, és az eredmények pontosabbak.

6.2 Módszerválasztó varázsló

A szoftver tartalmaz egy varázsló felületet, amely segíti a felhasználót a paraméterek hatásának megértésében és beállításában. Az alapértékek azonban nem alkalmazkodnak a szövegekhez, ezért készítettem el egy előzetes, pusztán heurisztikus úton működő elemzőt, amely javaslatot tesz a paraméterek beállítására a korpusz bizonyos tulajdonságainak vizsgálata alapján.

Habár, ahogy korábban is írtam, a stilometriában nem ismeretes általános, mindenre működő módszer a pontos paraméter-beállításra: egyszerűen túl sok tényezővel kell számolni, ha ilyesmivel próbálkozunk. Ugyanakkor vannak bizonyos jelenségek, amiket ki lehet használni: ezekre építettem az itt tárgyalt heurisztikus elemzőt, amely a szövegek statisztikai adatait figyelembe véve igyekszik egy becslést adni a legfontosabb paraméterek (Statisztikai módszerek fejezet) kezdőértékére.

Összesen öt paramétere van a klasszikus, statisztikai alapú módszereknek, amelyekről fentebb már részletesen volt szó. Ezek a következők: vizsgált n-gram típusa (szó vagy karakter), hossza, az MFW %-os értéke, a Culling %-os értéke és a felhasznált távolságmérési módszer. Ezek heurisztikus megközelítésű becsléseire az alábbi módszereket használtam fel.

6.2.1 N-gram típusa és hossza

Az n-gramok típus és hossza nagyban függ a nyelv jellegzetességeitől. Ennek automatizálása, bár elképzelhető, egy nagyon bonyolult és komoly nyelvészeti tudást feltételező feladat, amelyre ezen projekt keretein belül nem vállalkoztam. Szerencsére nem csak ez az egyetlen jellegzetessége a szövegnek, amely komoly hatással van rá: a szavak gyakorisága is fontos.

A szava gyakorisága, bár igen sok tényezőtől függhet, mégis ad egyfajta egyszerű következtetési modellt az n-gramra. Amennyiben a gyakoriság nagyon alacsony, érdemes szó n-gramok helyett inkább betű n-gramokat használni: egészen extrém (1% alatti) átlag esetén betű 3-gramokat javasol a szoftver.

Amennyiben a szavak átlagos gyakorisága 1% és 5% között van, ez pont megfelelő arra, hogy szimplán a szavak előfordulását használjuk, vagyis szó 1-gramokat használjunk az elemzésben. Ez a klasszikus módszer.

Ha a szavak átlagos gyakorisága 5% felett található, érdemes átváltani szó 1-gramokról 2-gramokra: a gyakori szavak miatt valószínűbb a kombinációk előfordulása, és azok sokkal jobb eredményt tudnak adni, ha statisztikailag releváns számban vannak jelen.

6.2.2 MFW

Az MFW (Most Frequent Words) értéke az egyik legfontosabb az elemzések során. Ennek megbecsülésére szükségem volt a szavak gyakoriságlistájára. Ezt szerencsére a stylo egyik függvénye hatékonyan képes prezentálni, így itt erre támaszkodtam. Az eredményből egyetlen számra van szükségünk: a szavak átlagos gyakoriságára. Azonban míg az előbbi esetben megfelelt azok relatív (az szöveg hosszához viszonyított) gyakorisága (amit a stylo magától prezentál), itt a pontos szám kell, ezért kell magamnak elvégezni a számítást.

Az MFW értéke az átlagos szógyakorisággal egyenesen arányos, százalékos érték. Ennek közelítésére a következő képletet használtam:

$$MFW = \frac{m}{m+5} * 100$$

Az m a kiszámolt átlagos gyakoriság. Ez a képlet alacsony m érték mellett alacsony MFW-t eredményez (ami jó, mert ilyenkor valószínűleg minden szó számít), míg magas m érték mellett magas MFW-t eredményez, de sohasem éri el a 100%-ot. A képlet jól működik, ha a korpusz nem tartalmaz extrémen sok vagy kevés szöveget.

6.2.3 Culling és lista levágása

A Culling értékének megbecsüléséhez sajnos a szavak gyakorisága nem használható fel. Itt sokkal inkább a korpuszban található szövegek viszonya az, ami a becsléshez hasznos számunkra. A Culling értékének annál nagyobbnak kell lennie, minél nagyobb különbségek vannak a fájlokban található szó-listák között. Ennek teljes feltérképezése sajnos igencsak számításigényes feladat, ezért egy egyszerűbb módszer került felhasználásra: a fájlok hosszából indulok ki. Minél rövidebb fájljaink vannak, annál alacsonyabb Cullingra lesz szükségünk, hiszen a magas érték jó eséllyel drasztikusan leredukálná a felhasználható szavak számát. Itt egy valamivel összetettebb képletre volt szükség:

$$Culling = \frac{m}{m + \left(\frac{100000}{l}\right)} * 100$$

Ahol az m az átlagos fájlhossz (szavakban), az l pedig a teljes korpusz mérete szavakban. Az l azért szükséges, mert kisméretű korpusz esetén a Culling-nak is alacsonyabbnak kell lennie, különben nem marad statisztikailag elegendő szó az elemzéshez. Ez a képlet jól működik közepes vagy rövid szövegekhez (kb. 50000 karakter hosszúságig).

Ezen felül, ha a gyakorisági lista hossza nagyobb, mint a szavak számának egytizede, akkor a szavak számának egytizedénél levágja a listát.

6.2.4 Távolságmérték

A távolságmértékeket alapvetően két csoportra osztottam: az „egyszerű” távolságmértékre és az „összetett” távolságmértékekre. Ebben az esetben az egyszerű jelenti a szövegek szerző szerinti eloszlástól nagyjából független módszereket, az összetett pedig azokat, amelyek számára fontos ez. Ezek alapján az elosztás a következő:

Egyszerű módszerek:

- Eder szimpla deltája
- Manhattan
- Canberra
- Euklideszi
- Koszinusz

Összetett módszerek:

- Burrows Deltája
- Argamon Deltája
- Eder Deltája

Az elemző megvizsgálja, hogy a szerzőkhöz tartozó szövegekből mennyi van (ezt, ahogy a stylo is, a szövegek elnevezéséből találja ki), és ezek után megvizsgálja, hogy van-e olyan írópár, ahol az egyik írónak több, mint másfélszer annyi szövege található a korpuszban, mint a másiknak. Ha igen, az egyszerű módszereket ajánlja. Ha nem, az összetetteket (lévén, ha a körülmények megengedik a használatukat, azok általában pontosabb eredményt adnak).

6.2.5 Mintavételezés

Az elemzés kitér a mintavételezésre is. Amennyiben a szövegek közt van olyan pár, ahol az egyik szöveg legalább kétszer olyan hosszú, mint a másik, javasolja a mintavételezést, egyébként nem. Fontos megjegyezni, hogy a javaslatok a varázslóba kerülnek, ahol a felhasználó a távolságmérték és a mintavételezés esetén választhat a javasolt csoportból.

6.3 Paraméter-optimalizáló

Habár a heurisztikus elemző gyors és megfelelő a laikusok segítségére, a pontossága elmarad egy, a stilometriában jártas felhasználó beállításai mögött. Ahhoz, hogy ténylegesen pontos, használható beállításokat tudjunk készíteni, ennél többre van szükségünk.

A megoldást az esetünkben egy jól ismert módszer, az előzetes paraméterezés jelentette, amely gyakran használatos szerzőiség-megállapítási feladatokra. Ennek során a kísérlet elvégzője

elsőként összeállít egy korpuszt, amely azon szerzőktől tartalmaz lehetőség szerint minél hasonlóbb hosszúsággal és témával, akik valószínűleg a szöveg szerzői lehetnek.

Ezek után ezen a korpuszon több kísérleti futtatást végeznek, különböző paraméterekkel. Kézzel folyamatosan módosítják őket, egészen addig, amíg a beállítások megfelelő mértékben elkülönítik a szövegeket klaszterekbe. Ekkor, az ily módon kikísérletezett paramétereket megtartva elvégezzük a kísérletet az ismeretlen szerzőjű szövegekre is. A gondolatmenet az, hogy ha ismert szerzőjű szövegekre jól működött, amennyiben a szöveg valamelyik szerzőhöz tartozik, a paramétereknek köszönhetően itt is egyértelműen meg fog mutatkozni, hova kerül.

Ez tehát az alapvető módszer, amit automatizáltam. Figyelembe kellett vennem, hogy a lépésszám nem lehet túl magas, főleg, ha mindennapi használatra is alkalmassá akarom tenni a rendszert: a stylo függvény egyszeri lefutása is igen hosszú, akár fél percig is eltarthat. Már egészen kevés, százas nagyságrendű lépés esetén is órákba telik.

A megoldáshoz végül a szimulált lehűlés algoritmusát választottam, a legfontosabb paraméterek lehetséges értékei által leírt térben. Ez az algoritmus jól paraméterezhető lépésszám terén, és jól bevált, megbízható megoldás. Sajnos önmagában ez az algoritmus nem megfelelő: amennyiben szimplán véletlenszerű kezdőpontot használok, a végeredmény ritkán pontos. Szükségünk van előkészítésre.

Ennek két módja van: az első, hogy a fentebb bemutatott heurisztikus elemzőt használom fel a kezdeti érték generálására. Ez meglehetősen jó eredményeket tud hozni, de bizonyos korpuszok miatt, ahol maga az elemző nagyon pontatlan eredményt ad, kénytelen voltam egy másik módszert használni. A módszerek függetlensége érdekében végül ez került bele a szoftverbe is: az elemzést megelőzően kiválasztok valahány pontot véletlenszerűen a keresési térben, és ezek közül kiválasztom a legjobbat. Onnan fog indulni a keresés.

Természetesen a fent leírt módszer azt is jelenti, hogy szükségem van egy számszerűsítésre, ami megmondja, hogy az adott paraméterezéssel elvégzett elemzés mennyire hatékony. Ez egy nemtriviális feladat, amelyre több kísérlet után végül a következő algoritmus született, mint megoldás:

Minden szövegre megvizsgálom, hogy az eredményként szolgáló táblában mennyire van közel az összes többi szöveghez. Ezeket a távolságokat összegezem a szöveg tényleges szerzőjére nézve és az összes többi szövegre nézve. Ha az összes többi szövegre nézve nagyobb eredményt kapok, a szöveg hibásnak lesz jelölve, egyébként a szöveg helyesnek lesz jelölve. A végén a hibás szövegek számából a helyes szövegek számát kivonva kapom a mérés pontosságát.

6.3.1 Paraméterek és értékkészletük

A futtatás során csak akkor kerül újra feldolgozásra a korpusz, ha szükséges, azaz változott az első két paraméter valamelyike. A keresés végén nem a végső állapot kerül felhasználásra, hanem a legjobb eredményhez tartozó kombináció. Fontos megemlíteni, hogy a keresés gyorsítása

érdekében egyrészt eltárolásra kerül minden paraméter-kombinációhoz a kiszámolt eredmény, másrészt vannak bizonyos megkötések az értékkészletekre nézve.

- Az n-gramra nézve a keresés egy egyszerűsítéssel él: amennyiben szavakat és nem karaktereket használunk, a felét veszi az n értékének, felfelé kerekítve. Egyébként az n értéke 2 és 6 között lehet, a típus értéke pedig karakter vagy szó.
- Az MFW értéke 5 és 100 között engedélyezett, 5%-os lépésközzel.
- A Culling értéke 0 és 90 között engedélyezett, 5%-os lépésközzel. Egy megkötés van rá: ha az n értéke nagyobb, mint 2, a Culling értéke nem lehet 30%-nál nagyobb, míg, ha az n értéke nagyobb, mint 4, a Culling értéke nem lehet 10%-nál nagyobb.
- A távolságmérték a nyolc távolságmérték bármelyike lehet.

7 Implementáció

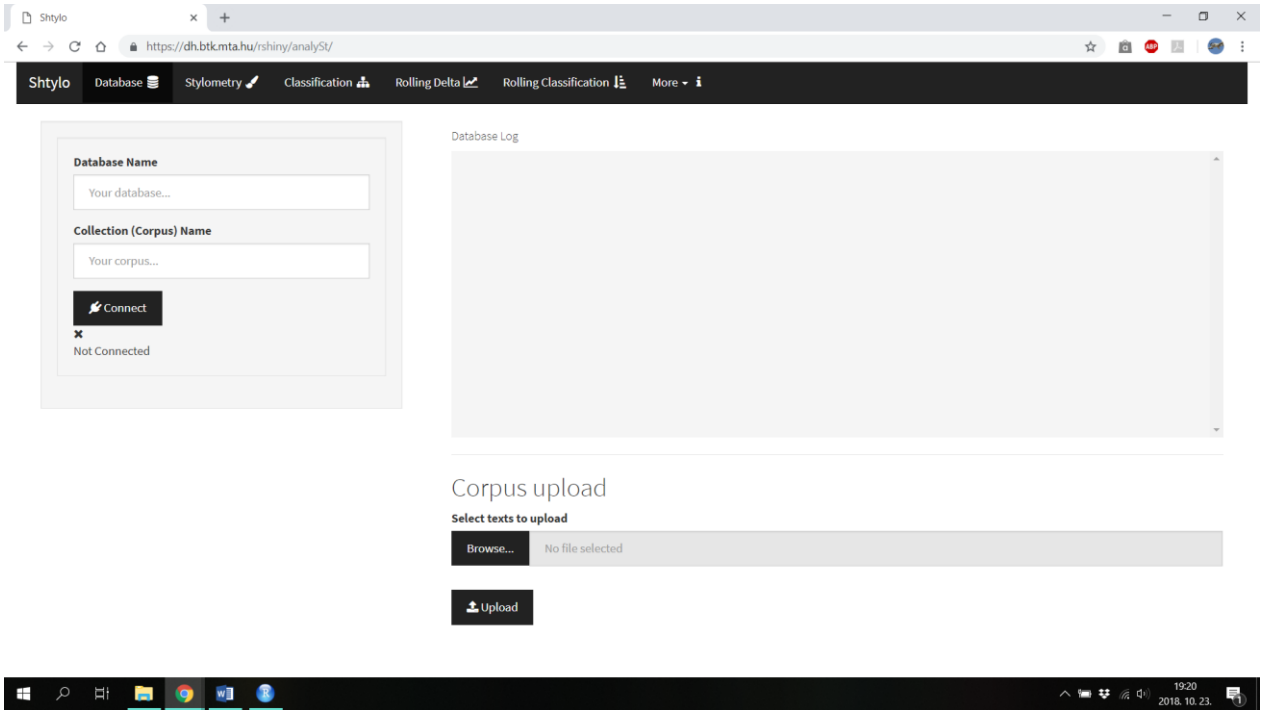
A rendszer gyakorlatilag a Shtylo továbbfejlesztése: megtartotta az alapvető struktúrát, és minden elemet, amely eredetileg benne volt. Ezeket felhasználva és újabb elemeket hozzáépítve készült el maga a dolgozat témájául szolgáló szoftver.

A Shiny architektúrája lehetővé teszi, hogy R-ben, ami nem egy kifejezetten objektum-orientált nyelv (sokkal közelebb áll a funkcionális nyelvekhez) is fejleszthessünk összetett, többrétegű alkalmazásokat. A Shiny project alapját három R fájl jelenti: az ui.R, a global.R és a server.R. Ezekbe kerül be (közvetlenül vagy importálással) minden kód, ami a működéshez szükséges. A global.R a függőségeket és globális beállítások betöltését tartalmazza, így azzal nem foglalkozunk a továbbiakban részletesen.

7.1.1 A GUI felépítése

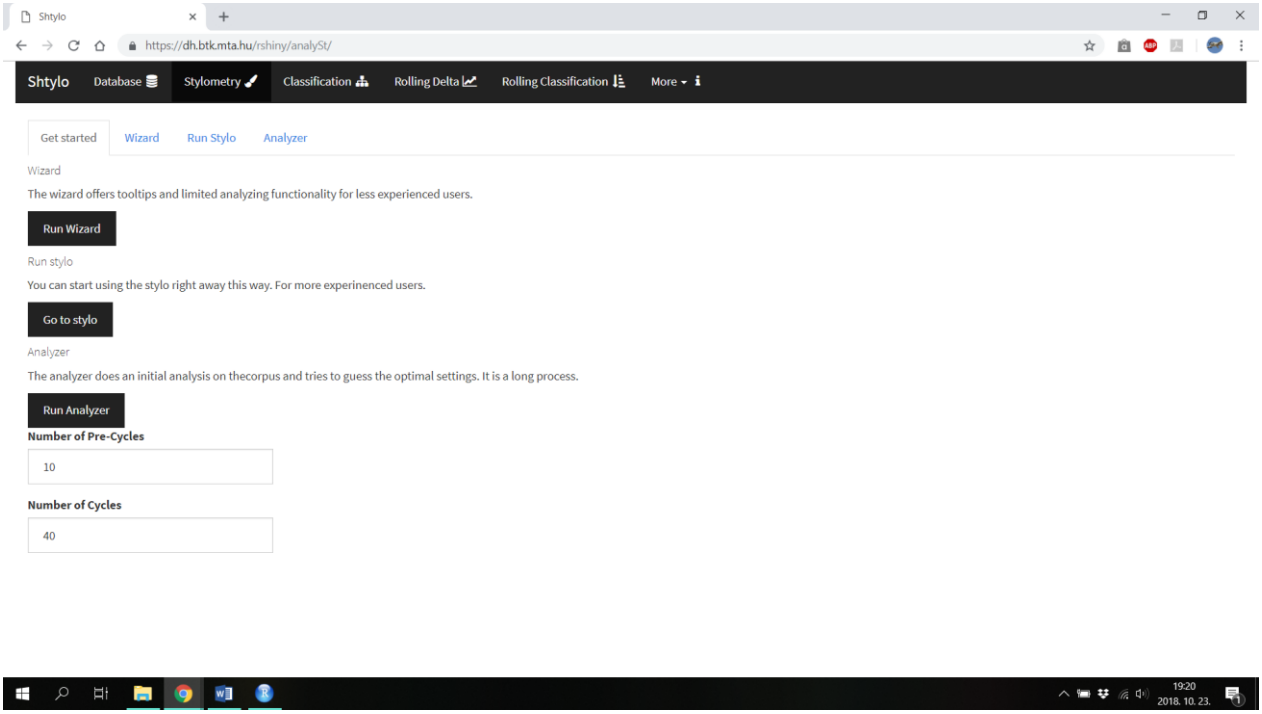
A GUI alapját az ui.R jelenti. Ez igazából csak felhasználja a root.view.R-t. A root.view.R tartalmazza az alapvető felépítését a GUI-nak, ebben találhatóak az egyes nézetek, amelyek a felület tetején található navigációs panel segítségével érhetők el. A kezdőoldalt a db.view.R fájlban definiált felület jelenti.

Ez a felület teljes egészében az eredeti Shtylo része, ezért itt csak röviden ismertetjük a működését. A felhasználó feltölthet korpuszokat adatbázisokba és azokon belül kollekciókba. A fájlok egyenesen a korpusz könyvtárba kerülnek a szerveren. A felhasználó legközelebbi látogatásakor ugyanazon korpusz elérését megadva képes ismételten használni a már korábban feltöltött korpuszokat.



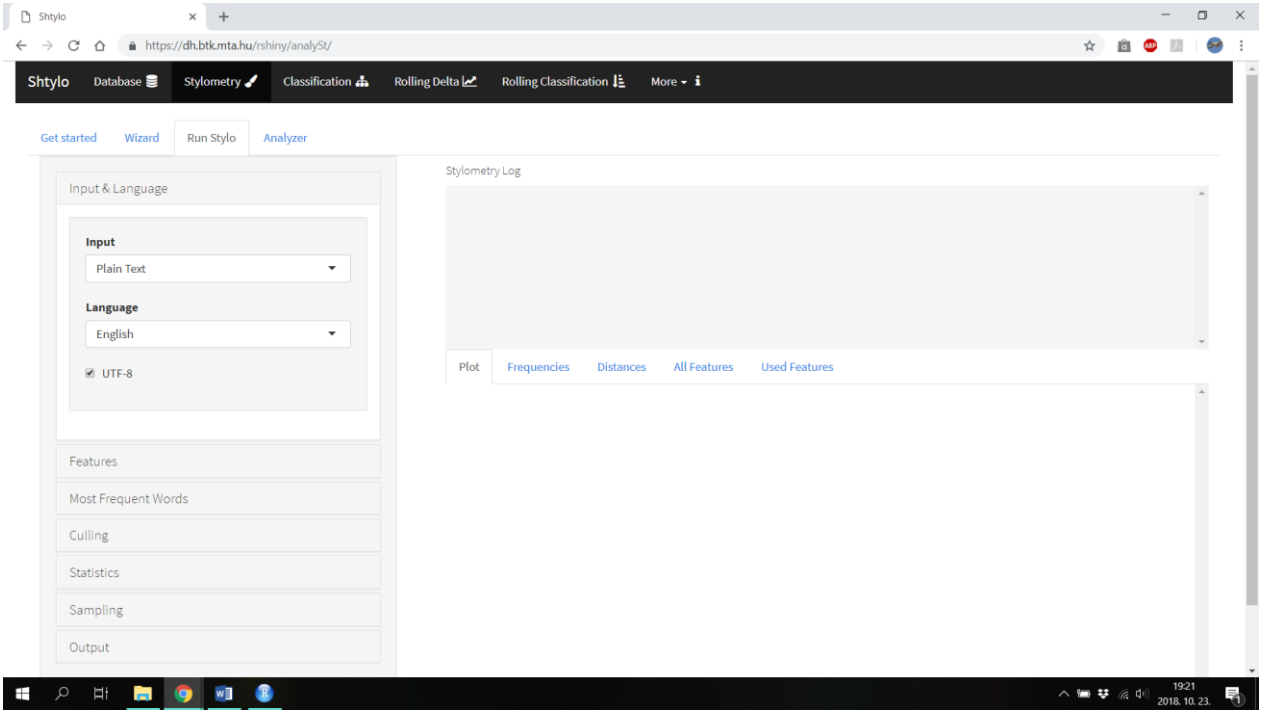
4. ábra: a korpusz-menedzser felület

A stilometria panel tartalmazza a stylo-hoz tartozó felületet az eredeti Shtylo-ban. Ez lecserélésre került: helyette immáron innen érhető el három elem: az eredeti felület (amely a stylo.main.view.R és a stylo.sidebar.view.R fájlokban van definiálva), a varázsló és az analizátor.



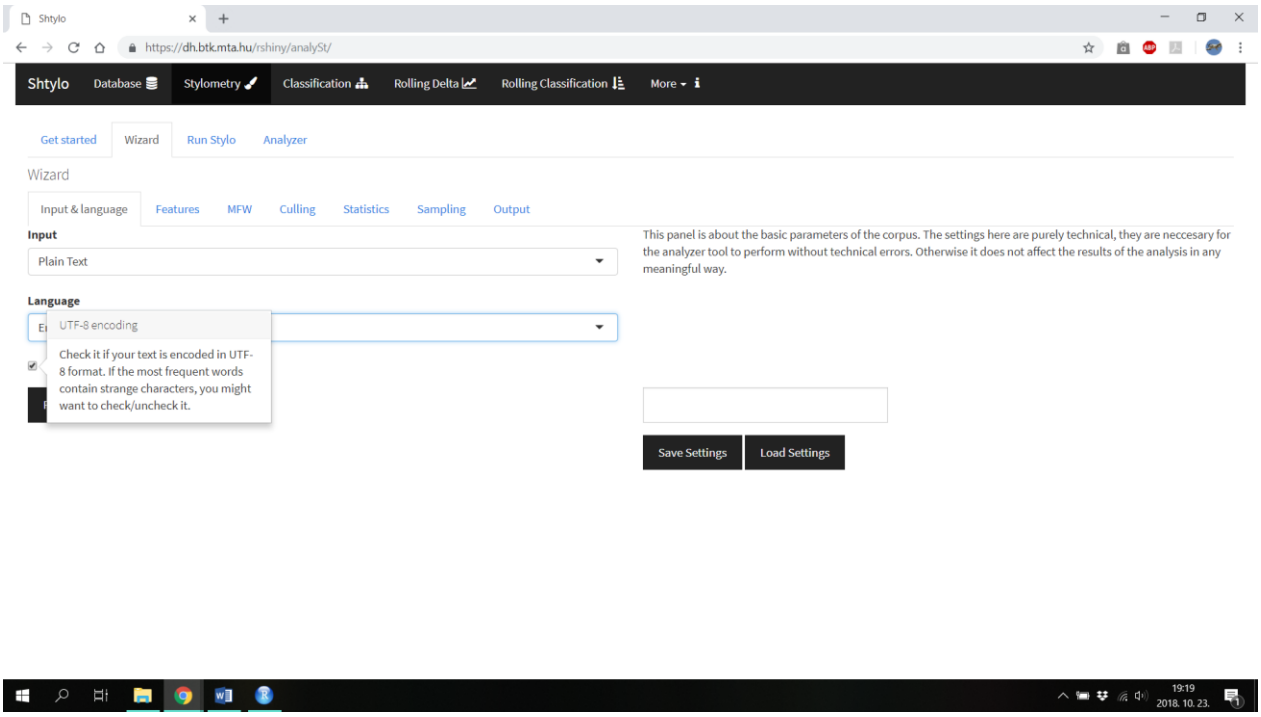
5. ábra: a kezdőoldal

Az eredeti felület a fájloknak megfelelően két fő részből áll: a main jeleníti meg az elemzés eredményét, míg a sidebar szolgál a beállítások megadására. Legördülő almenükből áll, amelyek nagyjából követik a stylo tematikus csoportosítását, és a fontosabb beállításoknak adnak csak helyet (több, kevésbé lényeges kényelmi funkció nem lett kivezetve a grafikus felületre a felhasználó összezavarásának elkerülése érdekében).



6. ábra: a Shtylo eredeti felülete

A varázsló felülete teljes egészében új fejlesztés: a beállítási opciók megfelelnek a Shtylo-ban leírtaknak, és ki vannak egészítve a felület alján három gombbal: Next, Previous, Apply. Az első kettő magától érthetődően a navigációra van, az Apply gomb pedig átviszi a beállításokat a régi felületre, hogy onnan elindítható legyen az elemzés. Ez lehetővé teszi a régi felület újrafelhasználását. Az MFW beállítására szolgáló felület segítségképp megjeleníti a 10 leggyakoribb szó listáját is.



7. ábra: a varázsló felülete a súgóval

A varázsló felület elérhető közvetlenül, a stilometria felület tetején található navigációs panellel, vagy a Wizard gomb megnyomásával. Ekkor egy rövid ideig a program várakozik. Ezt az okozza, hogy ilyenkor automatikusan lefut az előzetes elemzés heurisztikus verziója, amely a kereséses módszerhez képest rendkívül gyorsan lefut.

7.1.2 A szerver felépítése

A Shiny szerveroldala alapvetően session-ökre épül. A server.R betölti a kontrollereket, amelyek a felhasználói felületen történő változásokra reagálnak (gombnyomások stb.). Ezek egymásba is ágyazhatók, a fontos, hogy a megfelelő attribútumok át legyenek adva: ez elsődlegesen az input (amely a felhasználó bemenetét adja), az output, a session, a db.service (ami a MongoDB alapú adatbázishoz biztosít hozzáférést), és a log.service (ami értelemszerűen a logoláshoz kell). A kontrollerek fájljainak beolvasását a server.R-ben végezzük el, és ezeket használva építjük fel a fő kontrollereket: a sidebar kontrollert, a main kontrollert, és a két új kontrollert: a wizard-hoz és az analyzer-hez tartozókat. A log és a db egy-egy saját kontrollert is kapott, amelyek a szolgáltatásokhoz való hozzáférést biztosítják. Itt nem tárgyaljuk őket ennél részletesebben.

A sidebar kontroller és a main kontroller felelős a Shtylo eredeti felületének működéséért. Ezek mindössze minimális változtatásokon estek át. A sidebar kontrollere felelős a beállítások rész előkészítéséért és a helyes működés fenntartásáért. A main kontroller felelős a stylo meghívásáért a megfelelő paraméterekkel és a kimenethez való hozzáférés-biztosításért.

A wizard kontrollere tartalmazza a varázsló működéséhez szükséges szerveroldali részeket. A paraméterlistája tartalmazza a preanalyzer nevezetű függvényt, ez a heurisztikus elemző működését biztosítja. Erre szüksége van a varázslónak, ugyanis normál körülmények között lefuttatja azt.

A wizard kontrollere ezen felül felhasználja a stylo.wizard.load.R-ben és a stylo.wizard.save.R-ben tárolt eljárásokat, amelyek a megadott beállítások fájlba mentéséről és onnan való beolvasásáról gondoskodnak. A fájlba mentés az R egyszerű, beépített eljárásaival történik, ahol a beállításokból egy vektort képezünk, és azt szerializáljuk.

A stylo.wizard.preanalyzer.controller.R tartalmazza a heurisztikus elemzéshez tartozó funkciókat. Ez a komponens betölti a korpuszt és elvégez rajta bizonyos alapvető parsing feladatokat, de nem futtatja a stylo függvényt, azaz nem vizsgál eredményeket. Az eljárásba bele van építve az is, hogy az eredményeket automatikusan importálja a varázsló felületébe: pont emiatt igazából ez a rész a varázsló részének is tekinthető architekturális szempontból.

A stylo.wizard.update.R egy eljárást tartalmaz, amely elvégzi a varázslóban beállított adatok átmásolását az eredeti felületre a stilometriai elemzés elvégzéséhez. Annak, hogy nem egyenesen a futtatáshoz irányít a szoftver a varázsló végén, az az oka, hogy a varázsló a későbbi fejlesztések során is elsősorban egy egyszerűsített, laikus felhasználók számára készült felületként fog megjelenni, és amennyiben ehhez bizonyos beállítások lekerülnének a felületről, azon felhasználók számára, akik finomabb beállításokat akarnak, ily módon rendelkezésre áll a teljes funkcionalitás a régi felületen.

A stylo.analyzer.controller.R tartalmazza a fő újítást: a kereséses elemzőt. Ennek pontos működését lentebb tárgyaljuk, de egy alapvető áttekintést adunk. Az elemző felhasználja a stylo függvényt, futása pedig jelentős időmennyiséget igényel. Pont emiatt a működésére bizonyos beállításokat előre meg lenni az indítására szolgáló felületen (ami a stilometria felület kezdőoldala). Az ott található számok jelentőségét itt nem tárgyaljuk, de fontosak az elemzés szempontjából.

Az elemző, ahogy a heurisztikus verzió, szintén beimportálja a varázslóba az eredményként kapott paramétereket. A kereséses elemző annyiból intelligensebb, hogy figyelembe veszi a varázslóban alkalmazott nyelvbeállítást, mivel ez nagy jelentőséggel bír a stylo meghívásakor.

8 Kísérletek

A módszer eredményességének tesztelésére két kísérletet állítottam össze, egy saját készítésű, a tesztelés szempontjainak megfelelően összeállított korpuszt, és két, már más kutatásokban felhasznált korpuszt, melyekhez a kapott eredményeket összehasonlítottam az általunk kapott eredményekkel. A kísérletek során szerzőiség-megállapítási feladatot használtam: ez az, amit leggyakrabban használnak, és amihez a legtöbb publikáció születik.

Az elemzés alapvető módszertana a következő volt: elvégezem több futtatást az optimalizáló algoritmussal különböző kezdőpont- és iterációszámokra, és az így kapott paraméterezést használva Burrows Delta algoritmusára [9] támaszkodva kiértékelem az eredményt.

8.1 19. századi magyar levelezések

Az optimális korpusz a módszer tesztelésére egyenlő számú szövegeket tartalmaz egyes szerzőktől. A korpusz, amelyet elkészítettem, a http://deba.unideb.hu/deba/levelezes/ weboldalon összegyűjtött szerzők között történt válogatás eredménye. Az alábbi szerzők levelezését használtam fel: Berzsenyi Dániel, Döbrentei Gábor, Kazincy Ferenc, Márton József, Ráday I. Gedeon, Ráday II. Gedeon, Szentmiklóssy Alajos, Szentpéteri Erzsébet, Szentpéteri Judit, Szentpéteri Katalin.

Ezen tíz szerző műveiből leválasztottam egy tesztelésre szolgáló részt és egy betanításra szolgáló részt. A tesztelésre szolgáló rész két-két szöveget tartalmaz minden szerzőtől, míg a betanításra szolgáló szöveg tartalmazza a maradékot.

Kezdő-pontok száma	Iterációk száma	Vizsgált elem típusa	Vizsgált elemek száma	MFW	Culling	Távolság-mérték	Delta eredménye
manuális		karakter	6	85	15	Koszinuszos	62.5%
heurisztikus		karakter	3	51	45	Burrows Deltája	75%
1	10	szó	1	20	0	Manhattan	62.5%
5	10	karakter	5	85	5	Manhattan	67.5%
1	20	karakter	2	35	30	Eder egyszerű Deltája	62.5%
5	20	karakter	4	75	15	Koszinusz	75%
5	50	karakter	3	65	20	Euklideszi	75%
50	100	karakter	4	100	0	Koszinusz	82.5%

1. táblázat: a 19. századi magyar levelek korpusszal végzett kísérletek eredményei. Az első sorban látható egy laikus felhasználó által megadott beállításokkal történő kísérlet, a második sorban pedig a heurisztika (a varázsló) által javasolt beállítások eredménye. A 3-7 oszlopok a beállításokat, az utolsó pedig az eredményt tartalmazza

Az elemzés során több különböző iterációszámmal végeztem el a kísérletet. Mint látható, a kezdőpontok száma nagy hatással van az elemzés eredményére: egyetlen kezdőpont esetén nem kaptam olyan jó eredményt, mint több kezdőpont esetén, még ha nagyobb is az iterációszám.

A kísérletből az is látszik, hogy az iterációszám nem növeli szignifikánsan az eredményt, habár az alsó sor egyértelműen a legjobb eredményt produkáló. A heurisztikus paraméterezés meglepően jó eredményt produkál.

8.2 Német és angol regények

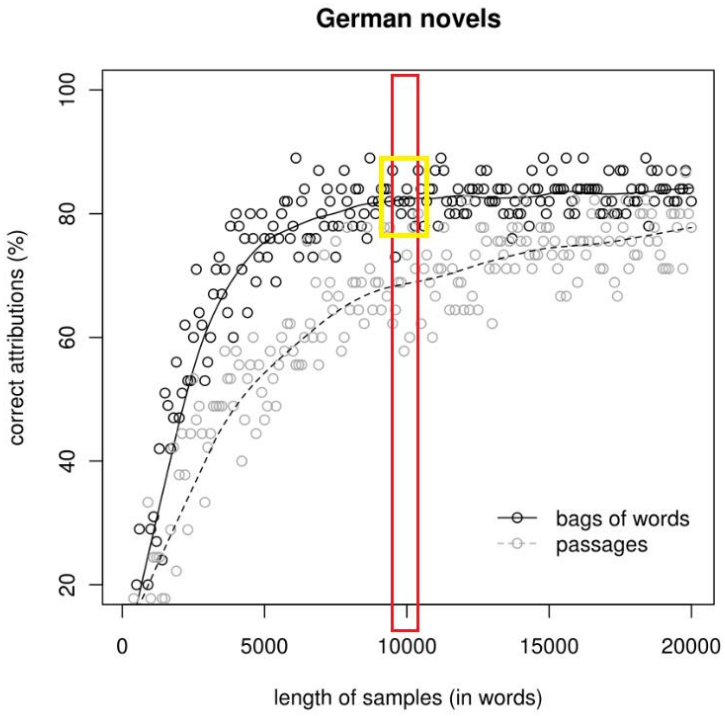
Maciej Eder egy, 2015-ben publikált kutatásában [12] olyan korpuszokat használ fel, amelyeket aztán később közzétett GitHub-on (https://github.com/computationalstylistics/). Ez lehetővé tette számomra, hogy a segítségükkel egy bejáratott, nyílt korpusz felhasználásával végezzek kísérleteket. A módszer a fentebb leírt volt: elsőként az optimalizáló algoritmus segítségével kinyertük az „optimális" paraméterezést, majd annak hatékonyságát teszteltem Burrows Delta nevezetű algoritmusával, amelyet a fent citált cikkben is használnak, és az eredményeket összehasonlítottam.

A kísérlethez használt korpusz abból a szempontból kevésbé volt megfelelő a kísérlethez, hogy kevés, de nagy hosszúságú szöveget tartalmazott: ez a kereső algoritmus szempontjából kedvezőtlen volt. A német nyelvű korpuszon végeztünk először kísérleteket. A korpuszt 45 tanító és 22 tesztelő szövegre osztottuk, melyből véletlenszerűen kiválasztottunk 5 szöveget a tesztelésre.

Kezdő-pontok száma	Iterációk száma	Vizsgált elem típusa	Vizsgált elemek száma	MFW	Culling	Távolság-mérték	Delta eredménye
nincs		karakter	3	84	97	Burrows Deltája	64%
1	10	karakter	2	40	70	Burrows Deltája	70.7%
5	20	karakter	4	90	15	Eder Deltája	72%
5	50	karakter	2	95	15	Eder egyszerű Deltája	78%

2. táblázat: a német regényekkel végzett kísérletek eredményei, az első sorban a heurisztika (a varázsló) által javasolt beállítások és az eredmény

Az eredményekből látszik, hogy a kezdőpontok és az iterációk számának növelése egyértelmű javulást mutat az eredmények tekintetében. A heurisztika itt elmaradt az algoritmikus megoldástól, de nem sokkal. Összehasonlításképpen álljon itt egy kép Eder publikációjából:



8. ábra: referencia a német regények korpusszal végzett kísérletekkel (forrás: [12])

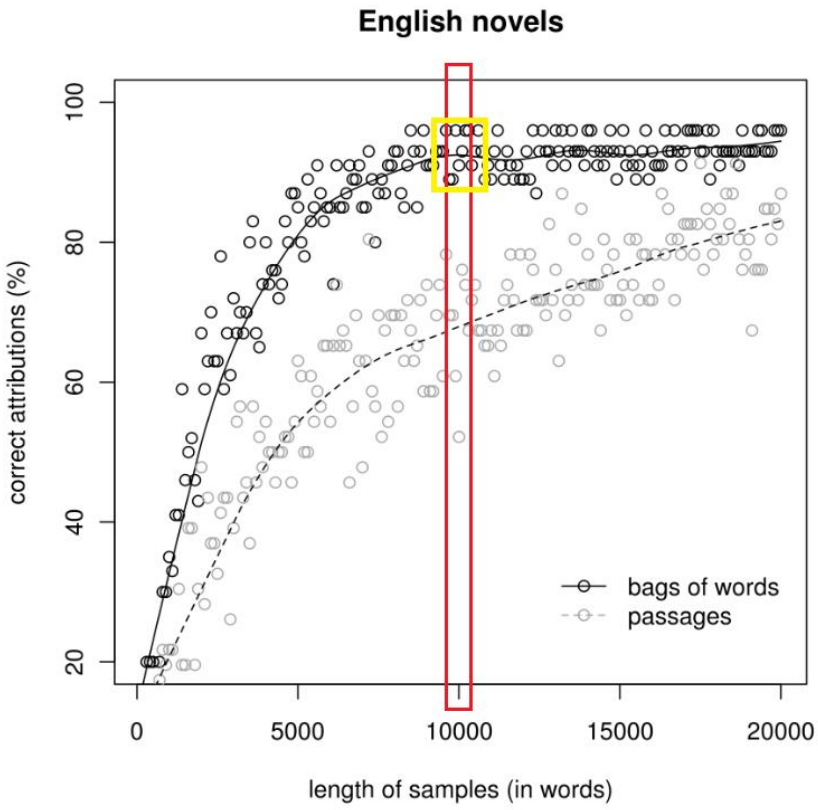
Látható, hogy az általunk használt 10000 szavas mintavételnél (piros keret) a „bag of words" módszer, amelyet mi is használunk, 75-90% közötti eredményeket jelent, körülbelül 80%-os átlaggal. Ezt a szintet sikerült megközelíteni, és valószínűleg magasabb iterációszámmal át is léphető. Az angol nyelvű korpuszon végzett kísérlet során a korpuszt egy 67 szövegből álló tanító szettre és egy 33 szövegből álló tesztszettre osztottuk, amikből 5-öt választottunk ki véletlenszerűen tesztelésre. Az eredményeket a következő táblázat foglalja össze:

Kezdő-pontok száma	Iterációk száma	Vizsgált elem típusa	Vizsgált elemek száma	MFW	Culling	Távolság-mérték	Ered-mény	Delta ered-ménye
nincs		karakter	3	97	99	Burrows Deltája	-	22.2%
1	10	karakter	2	50	15	Euklideszi	-61	50.8%
5	10	karakter	5	40	0	Burrows Deltája	-63	58.7%
5	20	karakter	6	90	5	Koszinusz	-61	61.4%
5	50	karakter	6	75	10	Canberra	-63	57.7%

3. táblázat: az angol regényekkel végzett kísérletek eredményei. Az első sorban a heurisztika (a varázsló) által javasolt beállítások és az eredmény

Az eredmény hasonló a magyar levelekkel végzett kísérlethez: az iterációk számának 20-ról 50-re növelése nem hoz javulást, sőt, apróbb romlást hoz. Ez valószínűleg nagyszámú kísérlet elvégzésével eltűnne, vagy legalábbis csökkenne. A heurisztika itt nagyon gyenge eredményt produkált, igaz, a szövegek hossza túlságosan nagy a pontos Culling becsléshez.

Összehasonlításképp álljon itt még egy ábra Eder publikációjából:



9. ábra: referencia a német regények korpusszal végzett kísérletekkel (forrás: [12])

Itt is pirossal van jelölve a 10000 szó hosszú mintavételezést használó kísérletek eredménye, és sárgával a releváns terület: az eredmények 85-95% között mozognak, 90%-os átlaggal. Ehhez képest az eredményeink elmaradnak.

Ennek elsődleges oka a korpusz jellegéből fakad: kevés szöveg szerzőnként, sok szöveg. Az algoritmusnak nem áll rendelkezésére kellő mennyiségű minta ahhoz, hogy hatékonyan tudjon optimalizálni. Mindezek mellett az eredmények javuló tendenciát mutatnak a kezdőpontszám és iterációszám növelésével (habár az utolsó eredmény alacsonyabb, de ez valószínűleg a kevés elvégzett kísérlet eredménye).

8.3 Eredmények

Összességében elmondhatjuk, hogy az algoritmus megfelelő paraméterbeállítást javasolt, amely alig marad el a terület egyik vezető kutatójának manuális beállításaitól, így jelentősen javíthatja a stilometriai elemzések használhatóságát. Javulást mutatott az iterációszám és a kezdőpontok számának növelésére, ami nagyobb számítási kapacitás és hosszabb futási idő mellett valószínűsíti még jobb eredmények elérhetőségét.

9 Összegzés, továbbfejlesztés

A megvalósított szoftver nagyban hozzájárul ahhoz, hogy megkönnyítse összetett szövegelemzési módszerek alkalmazását egyrészt azok paramétereinek heurisztikus beállításával vagy kereséssel történő finomhangolásával, másrészt a kidolgozott felhőalapú megoldás jelentősen megkönnyíti a kutatók munkáját otthonról vagy egyéb, nem nagy teljesítményű számítógépekről. Mind a súgó, mind a varázsló sokat segít a felület használatában, az architektúra pedig lehetővé teszi, hogy telepítés nélkül, pusztán egy böngészőből futtathatók legyenek a kísérletek.

Az elemző algoritmus és a heurisztika sikeresen felülmúlta a laikus felhasználó által elért eredményt, bár nem érik el a kutatók által végzett kísérletek eredményeit, megfelelő beállítások mellett megközelítették azokat. Várható, hogy továbbfejlesztés esetén jobberedményeket érjünk el velük akár az iterációszám növelésével, akár az algoritmus finomításával.

Az elsődleges továbbfejlesztési irány az algoritmus alapos vizsgálata. A keresés ugyan elméletileg hatékonyan bejárja az állapotteret, de nem vesz figyelembe semmilyen szabályszerűséget, sem összefüggést az egyes paraméterek között. Ezen összefüggések megvizsgálása és felhasználása nagymértékben javíthatja az elemzés eredményét. Ezen felül mindenképpen szükséges nagyobb lépésszám mellett kísérleteket végezni, mivel a fent szereplő lépésszámok relatíve alacsonyak. Fontos lenne vizsgálni a kezdőpont megválasztására használt módszert is, mert valószínűleg létezik hatékonyabb megoldás.

A fejlesztés másik fontos iránya a stylo könyvtár többi függvényének beépítése a szoftverbe. Ezek szintén fontos, hatékony eszközt adnak a felhasználó kezébe. Az ő beállításaik optimalizációja is egy érdekes továbblépési irányt jelent, melyet fontos lenne megvizsgálni.

10 Hivatkozások

[1] Mendenhall, T. C. „The characteristic curves of composition” Science, IX, 237–49. 1887

[2] Yule, G.U. „On sentence-length as a statistical characteristic of style in prose, with application to two cases of disputed authorship” Biometrika, 30, 363-390. 1938

[3] Yule, G.U. „The statistical study of literary vocabulary” Cambridge University Press. 1944

[4] Zipf, G.K. „Selected studies of the principle of relative frequency in language” Harvard University Press, Cambridge, MA. 1932

[5] Mosteller, F. & Wallace, D.L. „Inference and disputed authorship: The Federalist” Addison-Wesley. 1964

[6] Grant, T. D. „Quantifying evidence for forensic authorship analysis” International Journal of Speech Language and the Law, 14(1), 1 -25. 2007

[7] Chaski, C.E. „Who’s at the keyboard? Authorship attribution in digital evidence investigations” International Journal of Digital Evidence, 4(1). 2005

[8] Frantzeskou, G., Stamatatos, E., Gritzalis, S., & Katsikas, S. „Effective identification of source code authors using byte-level information” In Proceedings of the 28th International Conference on Software Engineering (pp. 893-896). 2006

[9] Burrows, J.F. „‘Delta’: A measure of stylistic difference and a guide to likely authorship” Literary and Linguistic Computing, 17(3), 267-287 2002

[10] Hoover, D. „Testing Burrows’ Delta” Literary and Linguistic Computing, 19(4), 453-475. 2004

[11] Hoover, D. „Delta prime?” Literary and Linguistic Computing, 19(4), 477-495. 2004

[12] Maciej Eder „Does size matter? Authorship attribution, small samples, big problem” Literary and Linguistic Computing, Volume 30, Issue 2, 167–182, 2015 június

[13] Greta Franzini, Mike Kestemont, Gabriela Rotari, Melina Jander, Jeremi K. Ochab, Emily Franzini, Joanna Byszuk and Jan Rybicki „Attributing Authorship in the Noisy Digitized Correspondence of Jacob and Wilhelm Grimm” Frontiers in Digital Humanities, 2018

[14] Maciej Eder, Jan Rybicki and Mike Kestemont: “Stylometry with R: A Package for Computational Text Analysis”, The R Journal Vol. 8/1, 2016 augusztus

[15] Mike Kestemont, Justin Stover, Moshe Koppel, F.B. Karsdorp Walter Daelemans: „Authorship Verification with the Minmax Metric”, KNAW, 2016 július

[16] Efstathios Stamatatos „A survey of modern authorship attribution methods” in: Journal of the Association for Information Science and Technology, Volume 60, Issue 3, 538-556. oldal, 2009 március

[17] Patrick Juola „Authorship Attribution” in: Foundations and Trends in Information Retrieval, Vol. 1, No. 3, 233–334. oldal, 2008 március 7.

[18] Jack Grieve „Quantitative Authorship Attribution: An Evaluation of Techniques” in: *Literary and Linguistic Computing* Volume 22, Issue 3, 251–270. oldal, 2007 szeptember 1.

[19] Efstathios Stamatatos, Moshe Koppel: „Plagiarism and authorship analysis: introduction to the special issue", Lang Resources & Evaluation 45:1–4, 2011

[20] Maciej Eder: „*Style-Markers in Authorship Attribution A Cross-Language Study of the Authorial Fingerprint",* Studies in Polish Linguistics 6, 2011

[21] Jan Rybicki, Maciej Eder: „Deeper Delta across genres and languages: Do we really need the most frequent words?" LLC. 26. 315-321. 10.1093/llc/fqr031. 2011

[22] Piasecki, M., Walkowiak, T. and Eder, M. „Open stylometric system WebSty: Integrated language processing, analysis and visualisation" *Computational Methods in Science and Technology*, 24(1): 43–58, 2018

[23] Argamon, S. Interpreting Burrows' Delta: Geometric and probabilistic foundations. Literary and Linguistic Computing, 23(2), 131-147. 2008

[24] Lance, G. N.; Williams, W. T. "Computer programs for hierarchical polythetic classification ("similarity analysis")". *Computer Journal*. 9 (1): 60–64. 1966