A TUDOMÁNY, AZ OKTATÁS ÉS A KÖZGYŰJTEMÉNYI KISZOLGÁLÁS ÚJ INFORMATIKAI SZINERGIÁI

NETWORKSHOP 2026
35. Országos Informatikai Konferencia

2026. március 31–április 2.
Debreceni Egyetem, Debrecen

Szerkesztette: Tick József, Kokas Károly, Holl András

HUNGARNET Egyesület
Budapest, 2026

HUN-REN
Magyar Kutatási Hálózat

NETWORKSHOP

Szerkesztette: Tick József, Kokas Károly, Holl András

Tipográfia és tördelés: Vas Viktória
Korrektúra: Danyi Melinda
Angol nyelvi lektor: Lukács Katalin

Networkshop 2026 konferencia előadásainak közleményei
Debreceni Egyetem, Debrecen
2026. március 31–április 2.

ISBN 978-615-6792-29-7
DOI: https://doi.org/10.31915/NWS.2026

Kiadja a HUNGARNET Egyesület
az MTA Könyvtár és Információs Központ közreműködésével
Budapest
2026

Borítókép: freepik.com

DOI: 10.31915/NWS.2026.25

AZ ÖSSZETETT MEZŐK STRUKTURÁLT ADATAINAK SZÉTVÁLASZTÁSA AZ ÁLLAMI ANYAKÖNYVEKBEN

SEPARATING STRUCTURED DATA IN THE COMPLEX FIELDS OF HUNGARIAN CIVIL REGISTERS

Szűcs Kata Ágnes szucs.kata.agnes@mnl.gov.hu
Varga Emese varga.emese@mnl.gov.hu
Vadász Noémi vadasz.noemi@mnl.gov.hu
Záros Zsolt zaros.zsolt@mnl.gov.hu
Szatucsek Zoltán szatucsek.zoltan@mnl.gov.hu

Absztrakt

A kézírással készült adatok kinyerése szkennelt anyakönyvekből az összetett mezők miatt is kihívást jelent: egy mező többféle adattípust tartalmazhat, hiányozhat a fejléc és változhat a sorrend, miközben a HTR-kimenetek soralapúak. Tizenhét összetett mezőt azonosítottunk; a szabályalapú módszerek csak egyszerű esetekben működtek, ezért bevezettünk mezőre szabott LLM-megoldásokat prompt- és paraméterhangolással. Egyes modellek megbízható eredményeket adtak, másoknál adatvesztés, ID-kiesés, felesleges tokenek vagy instabilitás jelentkezett, amelyeket szigorú rendszerpromptokkal és prompt módosítással kezelünk. Reprezentatív tesztkorpuszt és mezőszintű mérőszámokat építettünk. Következtetés: nincs univerzális megoldás – néhány mező teljesen automatizálható, soknál maradnak bizonytalanságok; a legjobb eredmény a feladat részekre bontásából és a célzott LLM-ek szabályalapú ellenőrzésének kombinálásából adódott.

Kulcsszavak: állami anyakönyvek, kézírás-felismertetés (HTR), nagy nyelvi modellek (LLM), szemantikus szegmentálás, adatnormalizálás, Magyar Nemzeti Levéltár

Abstract

Extracting handwritten data from scanned registry books is challenging due to complex fields that may contain multiple data types, missing headers, and variable ordering, while HTR outputs are row-based. We identified 17 complex field types; rule-based methods worked only for simple cases, so we introduced field-specific LLM solutions with prompt and parameter tuning. Some models were stable; others showed data loss, ID dropout, superfluous tokens or instability, which we have mitigated by strict system prompts and tuning. We built a representative test corpora and field-level metrics. Conclusion: there is

no universal solution – some fields can be fully automated, many remain ambiguous; best results arise from decomposing the task and combining targeted LLMs with rule-based validation.

Keywords: civil registers, handwritten text recognition (HTR), large language models (LLM), semantic segmentation, data normalization, National Archives of Hungary

1. Bevezetés

A Magyar Nemzeti Levéltár (MNL) egyik legjelentősebb digitális bölcsészeti vállalása az 1895 utáni állami anyakönyvi másolatok átfogó, automatizált és mesterséges intelligenciával támogatott feldolgozása. Az MNL őrizetében lévő, mintegy 22 millió oldalnyi születési, házassági és halálozási anyakönyv nem csupán hatalmas adatmennyiséget képvisel, hanem a történeti demográfiai kutatások megkerülhetetlen forrásbázisát is alkotja.

A projekt stratégiai célkitűzése a kézzel írt adatok kinyerése a strukturális információkkal együtt, lehetővé téve a rekordok névtérhez kapcsolását és az entitások összekapcsolását. Jelen tanulmány a nagy nyelvi modellek (Large Language Models, LLMs) alkalmazására tett kísérletek eredményeit foglalja össze az úgynevezett összetett mezők szemantikai szétválasztásában és tisztításában.

1.2 Technológiai környezet

A nagy nyelvi modellek (LLM) futtatása támasztotta számítástechnológiai igényeket az MNL más intézményekkel együttműködésben tudja kielégíteni. A számításigényes munkafolyamatokat a HUN-REN Wigner Kutatóközpont[1] által biztosított, dedikált felhőalapú kutatási infrastruktúra szolgálja ki. A tesztelési fázis során az alábbi hardveres és szoftveres beállítások álltak rendelkezésre:

- **Számítási teljesítmény:** 4 db NVIDIA A100 40GB GPU egység, amely elengedhetetlen a nagy paraméterszámú modellek futtatásához.
- **Adattárolás:** 520 GB SSD tárolókapacitás.
- **Biztonság:** Titkosított, távoli elérésű szerverkörnyezet, amely garantálja a levéltári adatok integritását a számítási folyamatok alatt.

2. Munkafolyamat és módszertan

A teljes anyakönyv-feldolgozási lánc kilenc egymásra épülő lépésben foglalható össze, amelyek a digitalizálástól a publikálásig kísérik végig a folyamatot. Ebben a folyamatban az összetett mezők esetében a nagy nyelvi modellek hidat képeznek a kézírás-felismertetés (Handwritten Text Recognition, HTR) nyers kimenete és a strukturált adatbázisrekordok között.

1 https://wignerdc.wigner.hu/home (Utolsó letöltés: 2026.04.07.)

I. Előkészítési szakasz:

1. **Képek előfeldolgozása:** Digitális tisztítás és orientáció javítása.
2. **Sablonkiválasztás:** A megfelelő oldalképformátum (layout) azonosítása.
3. **Szegmentálás:** A kitöltött rovatok geometriai elhatárolása a képen.

II. HTR és utófeldolgozási szakasz

4. **HTR (kézírás-felismerés):** Szöveghipotézisek generálása.
5. **Adatbázisba töltés:** Ebben a fázisban kerülnek rögzítésre a nyers HTR-adatok a későbbi szemantikai tisztítás előtt.
6. **Adatnormalizálás és javítás:**

   6.1 Szabályalapú utófeldolgozási eljárások: a) hipotézis kiválasztás a legmagasabb konfidencia érték alapján, b) normalizálás (pl. leány → nő), c) javítás (pl. rom. katólikus → római katolikus)

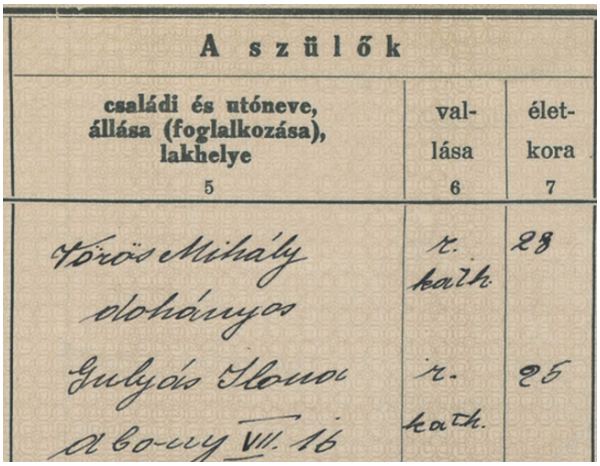
   6.2 LLM-alapú adatszétválasztás és helyesírási korrekció.

III. Entitásösszekapcsolási és -publikálási szakasz

7. **Adatok névtérhez kapcsolása:** Névtér azonosítók hozzárendelése a földrajzi nevekhez.
8. **Személyek összekapcsolása:** Rekord-összekapcsolás a későbbi családfa-rekonstrukcióhoz.
9. **Publikálás:** Az adatok indexelése és kereshetővé tétele a végfelhasználók számára.

2.1 Az összetett mezők kihívásai

Az összetett mezők kezelésének kérdése a teljes anyakönyv-feldolgozás komplex munkafolyamatának része. A „sima" és az „összetett" mezőket a cellában megjelenő adatok mennyisége mentén különböztetjük meg. Abban az esetben, amikor az anyakönyvvezetőnek több különböző típusú információt kellett beírnia egy-egy cellába, amelyek egy vagy több személyre is vonatkozhattak, összetett mezőkről beszélhetünk (pl. a szülők neve, állása, lakhelye vagy a menyasszony neve, foglalkozása, vallása, életkora és lakhelye).



A szülők		
családi és utóneve, állása (foglalkozása), lakhelye 5	vallása 6	életkora 7
Kiss Mihály dohányos	r. kath.	28
Gulyás Ilona a Bo-ucy VII. 16	r. kath.	25

1. ábra: Szülők neve, állása, lakhelye – egy összetett mező adatai

Míg az egyszerű – de akár egyes összetett mezők esetében is (pl. 1. ábra szülők vallása, szülők életkora) – szabályalapú utófeldolgozás is elegendő, a komplex eseteket nem tudjuk pusztán szabályalapon kezelni. A korábban külön lépésként lefutó hipotéziskiválasztás, adatbázisba töltés, illetve a normalizálás és szemantikai szétválasztás ezeknél egyetlen fázisban történik, még összetettebbé téve a feladatot.

Az összetett mezők (pl. „anyja neve, állása, lakhelye") feldolgozása az alábbi problémák miatt igényel fejlett nyelvi modelleket:

- **Határolójelek hiánya:** A különböző adatok (pl. személynév, foglalkozás) gyakran írásjelek nélkül követik egymást.
- **Szemantikai szegmentálás:** Az entitások jelentés szerinti elkülönítése az azonosításhoz (pl. szabó és Szabó).
- **Fejléc-következetlenség:** A cella tartalma gyakran eltér a fejlécben előírt adatoktól (pl. életkor helyett születési dátum megadása, családi állapot feltüntetése).
- **Szemantikai szegmentálás és entitássorrend inkonzisztenciája:** A kitöltő személy tetszőleges sorrendben rögzíthette az adatokat (pl. lakhely megelőzi a nevet, a családi állapot kerülhet a név elejére és a végére is, bizonyos adatokat kihagy, illetve nem ír le kétszer vagy épp megdupláz).

2.2 LLM-alapú feldolgozás: metodika

A kutatás során a modellek tesztelése három fázisban történt: (1) egyetlen komplex mezőn több modell mérése a „best-of" modell kiválasztásához; (2) a nyertes modell tesztelése több mezőtípuson; (3) modell-specifikus tesztelés dedikált mezőcsoportokon.

A modelleket finomhangolás nélkül és a reprodukálhatóság és visszamérhetőség érdekében rögzített paraméterekkel alkalmaztuk: seed=42 és hőmérséklet= 0,4.[2] Az első fázisban vizsgált modellek technikai jellemzőit az 1. táblázat foglalja össze.

Készítettünk egy benchmark korpuszt a tesztelt összetett mezőkre, amelyhez kétfajta átírást állítottunk elő, hogy kiderüljön, a modell mennyire távolodik el a hipotézisektől a normalizálás felé.

Modell	Méret	Kontextus
llama3.3:latest	43GB	128K
gemma3:27b	17GB	128K
qwen3:235b	142GB	256K
gpt-oss:120b	65GB	128K

1. táblázat: Az anyja neve, állása, lakhelye mezőn tesztelt modellek

A mérések kiterjedtek a **betűhív és szöveghű átírásra,** valamint a **sima** és a **transzformált** értékek számítására is. Utóbbi során az alábbi szabályokat érvényesítettük: kisbetűs konverzió, írásjelek és felesleges szóközök eltávolítása, valamint regex-alapú normalizálás (pl. cz → c mapping, kivéve a személynevekben, illetve a rövidítések feloldása: róm. kath. → római katolikus).

Az adatfeldolgozás során az LLM megkapja bementként a nyers HTR-hipotéziseket, majd ebből strukturált JSON formátumú kimenetet generál, melyben az összetett mezőket kulcs-érték párokra bontja a modell (pl. anya_nev: ”adat”, anya_foglalkozas: ”adat”, anya_lakhely: ”adat”).

2 Finomhangolás nélkül a modell csak a gyárilag előre betanított súlyokat használja; nem alkalmaztunk további feladatspecifikus finomhangolást (fine-tuning). A viselkedése így általánosabb, kisebb a pontosság, azonban az esély is a túlillesztésre. A seed bármely tetszőleges egész szám lehet; ugyanazzal a seed értékkel (és azonos környezettel) ugyanazt a pseudovéletlen sorozatot és így azonos generált kimenetet kapjuk. Tehát biztosítja a reprodukálhatóságot a generálás során előforduló véletlenszerű folyamatokhoz. A temperature (hőmérséklet) egy olyan kapcsoló, amely modell kockázatvállalását szabályozza a szöveggenerálás során. Az értéke 1 alatt kicsinek számít, a modell gyakran a legvalószínűbb szavakat választja, melynek következtében kevésbé változatos, stabil, „konzervatív” szöveget generál. 1-re állítva az alapviselkedése szerint működik, 1-nél magasabb érték esetén pedig több szó kap hasonló esélyt, tehát változatosabb, kreatívabb, de néha kevésbé koherens szöveget eredményezhet.

3. Eredmények, kiértékelés

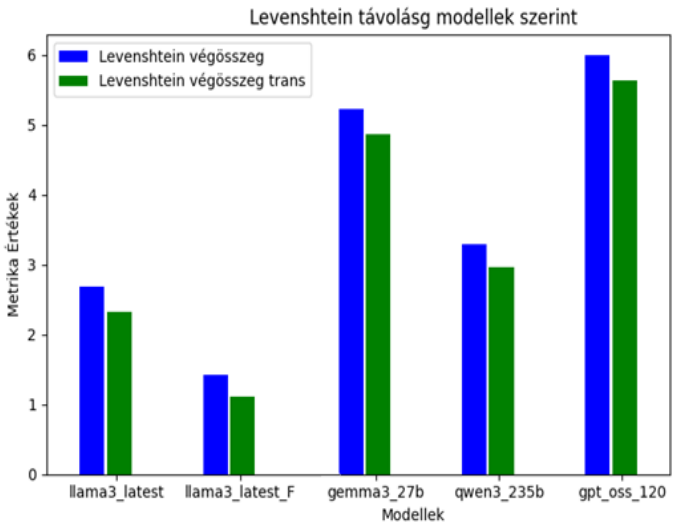
A teljesítmény értékelése mezőnkként egy 300 mintás véletlenszerű benchmark korpuszon alapult, Levenshtein-távolság[3] mérésével.

(1) A tesztelés első fázisában az anyja neve, állása, lakhelye mezőn tesztelt modellek közül – egy prompt finomhangolást követően (F) – a llama3.3 került ki legjobbként, átlagosan 1,4 értékű karaktereltéréssel. Az átlag Levenshtein-távolságok összege 2,7-ről 1,4-re csökkent, míg a transz értékek esetében 2,3-ról 1,1-re.

A 2. ábráról leolvasható, hogy a transzformált értékek minden esetben ugyanolyan mértékben alacsonyabbak, ami igazolja, hogy a normalizálási szabályok sikeresen távolítják el a stilisztikai zajt, amely nem tartalmi hiba.

Az eredmények további pontosítása végett a távolságmetrika mellett azt is vizsgáltuk, hogy mennyi volt a pontos egyezések, a hallucinációk és a kihagyások száma az összes értékhez viszonyítva. Ezeket a mutatókat mezőnkként generáljuk egy script segítségével.

Az anyja állása, lakhelye, neve mezőnél azt látjuk, hogy a 906 minta esetében 36 eltérés mutatkozik az elvárt kimenet és a modell által generált eredmény között. Ebből 6 esetben (0,6%) történt **hallucináció**, amikor a mérési alap (GT) üres volt, de az LLM adatot generált. Ezzel szemben 30 esetben (3,3%) fordult elő **kihagyás** (omission), ahol a modell nem adott választ a létező mérési alap mellett.



2. ábra: Levenshtein-távolságok betűhív átírás esetén az anyja neve, állása, lakhelye mező esetében.

3 A nyelvészetben és az informatika területén a Levenshtein-távolság egy karakterlánc-metrika, amely két sorozat közötti különbséget méri. Két szó közötti Levenshtein-távolság az a legkevesebb egykarakteres módosítás (beillesztés, törlés vagy csere), amely ahhoz szükséges, hogy az egyik szót a másikká alakítsuk.

Az eddigi pozitív eredmények mellett azonban figyelembe kell venni, hogy a pontos egyezések aránya (betűhív transzértékkel számolva) 39%. Ezen minták esetében tehát nincs szükség további beavatkozásra. Ehhez még a 3 alatti Levenshtein-távolsággal rendelkező mintákat is hozzátehetjük, ami további 26% -kal növeli az egyezések arányát, amely így 65%-ra egészül ki. A kimenetek 35%-a, tehát 4 és afölötti Levenshtein-távolságra van a benchmark korpuszban lejegyzettektől.

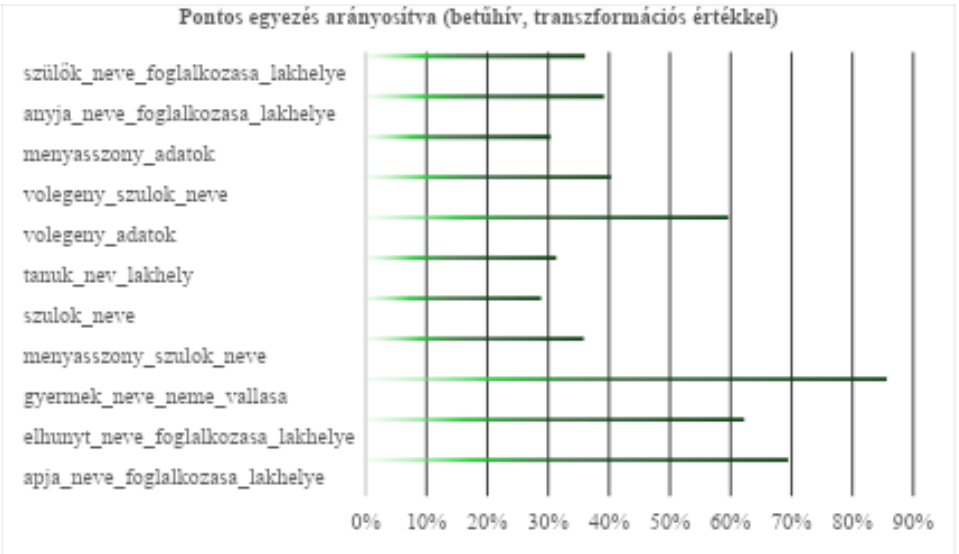
Ebből is látható, hogy egy-egy eredmény értelmezése nem annyira egyértelmű és sok számolást és gondolkodást igényel, amelyre azonban szükség van, ha a jövőbeni munkát és költségeket szeretnénk felmérni.

(2) A tesztelés második fázisban a többi mezőn is leteszteltük a llama 3.3 modellt. Ahogy az a 3. ábrán is látható, a pontos egyezések aránya széles kilengést mutatott.

Az eredmények tüzetes átvizsgálása során kiderült, hogy egyes mezőstruktúrák és adattípusok esetén sokat ront a modell, míg más esetekben könnyebben oldotta meg a feladatot.

A hibatípusok között a kihagyás több okból is előfordult. Az LLM rosszul kezelte a nagyon hosszú sorokat, egy név esetében – például, „néhai özvegy Bankó Jánosné született Halmos Sarolt," – a született utáni részt már lehagyta.

Előfordul olyan is, hogy az LLM rosszul szegmentálja a sztringeket – például egy hibás HTR-kimenet értelmezése esetén[4] – és nem ismeri fel, hogy az adott szövegrész nem a foglalkozáshoz, hanem a lakhelyhez tartozik. Ebből kétféle hiba is eredhet: egy kihagyás, és egy rosszul felcímkézett adat.



3. ábra: Llama3.3 teljesítménye az összetett mezőkön

4 Mivel a pipeline lépései egymásra épülnek, ezért minden korábbi folyamat teljesítménye kihatással van a végső eredményre.

Figyelembe véve az intézmény célkitűzését, egy anyakönyvi adatbázisban a legrosszabb hibatípus a hallucináció, tehát az, amikor LLM kitalál egy adatot ahelyett, hogy a megadott bemeneti hipotézisekből dolgozna.

Kiszámíthatatlanul működik a modell a helyesírási hibák javítása és az írott formák normalizálása esetében. Az alábbi prompt (utasítás)[5] mellett csak bizonyos esetekben modernizálta a helyesírást, több alkalommal őrizte meg az eredeti cz-s formákat.

(3) A harmadik fázisban csoportokba osztottuk a mezőket az alapján, hogy milyen típusú adatok fordulhatnak elő bennük (példák ld. Függelék). Kiemelt figyelmet kaptak a három különböző típusú adatot magukban foglalók, például az „anyja/apja/elhunyt neve, foglalkozása, lakhelye", illetve a több különböző adatot tartalmazók, mint a „menyasszony/vőlegény adatai", valamint a „tanúk neve és lakhelye" típusú komplex mezők. Ezekben ugyanis magas az előforduló személynevek aránya, amelyek még a lakcímben is előfordulhatnak.

Ebben a fázisban újabb modellekkel kísérleteztünk, annak reményében, hogy jobb eredményeket kapunk az összetettebb, illetve vegyes adatokat tartalmazó mezőkkel.

A qwen3.5:122b-a10b[6] indításakor olyan hibákba ütköztünk, amelyek hosszútávon megakadályozták a modellel való munkát: futtatás közben leállhat, vagy kihagyja a képekhez tartozó ID-kat, amelyek egy későbbi adatbázisba visszatöltés esetén elengedhetetlenek.

A qwen3-235b:latest[7] esetében problémát jelentett a folyamatosan jelenlévő „gondolkodás"[8], amelyre sem a think=False és a /set nothink kapcsolók, sem a system prompt módosítása sem volt hatással. Ez a gondolkodó viselkedés hosszútávon költség- és időigényes, a plusz generált tokenek miatt.

A system prompt egy speciális kezdeti utasítás vagy kontextus, amelyet a futtatási környezet rögzít a modellnek. Célja, hogy keretezze a modell viselkedését és szerepét a teljes beszélgetés során. Meghatározza a modell hangját, stílusát, tiltott/engedélyezett viselkedést és magas szintű szabályokat esetleg szerepmeghatározásokat rögzít. A modell számára látható, de általában a felhasználó nem látja közvetlenül. Az általunk használt utasítás a következő volt: „You are a specialized JSON generator. Output ONLY raw JSON. No conversational text, no markdown code blocks, no explanations."

Mivel nem találtunk olyan beállítást, amellyel sikerült volna lekorlátozni ezt a viselkedést ezért a feladat összetettségét csökkentettük. A kezdeti prompt szöveget úgy módosítottuk, hogy az eredeti elképzeléshez képest ne válasszon ki legjobb hipotézist, hanem az összes sort címkézze annak megfelelően, hogy milyen adatot tartalmaz. Például:

---

5 Végül javítsd az elgépeléseket a címkékben. A keresztneveket bátrabban, a vezetékneveket kevésbé. [prompt részlet]

6 https://ollama.com/library/qwen3.5:122b-a10b (Utolsó letöltés 2026.05.08)

7 https://ollama.com/library/qwen3:235b (Utolsó letöltés 2026.05.08)

8 Internális gondolat/elemzés tageket generál és ír ki, mielőtt a feladatot ellátná.

```
A feladatismertetés után kötegelten küldök mezőket. Minden mező
egy többsoros mező a szülők adataival.
Címkézd a mezők tartalmát.

A címkézés szabályai:
1. tanu_1_nev, tanu_1_lakhely, tanu_2_nev, tanu_2_lakhely címke
lehet.
2. nem biztos, hogy minden címkéhez van adat.
3. a sorokban lévő listák a sor különböző hipotézisei. Minden
sor összes hipotézistét tartsd meg a címkében, az eredeti for-
mában listázva, pl. „1. sor”: [„MátÉ”, „Máté”, „Mátén”, „Má-
tés”, „Mátó”], „2. sor”: [...]
4. a mezőben egy címke egy vagy több egymást követő sorból áll
össze. Az összetartozó sorokat a hipotézisekkel együtt tedd egy
címkébe.
5. ha előfordul a listában a „-” és a „¬” karakter, az elvá-
lasztást jelent, jelezve, hogy az adott sorban lévő hipotézisek
összetartoznak a következő sorban lévőkkel.
6. a személynevek elemei között logikai kapcsolat van: tanu1_
nev: első vezetéknév + első keresztnév, tanu2_nev: második ve-
zetéknév + második keresztnév.
6.1. A vezetéknevek előtt szerepelhetnek özv., özvegy, néh.,
néhai előtagok. Ezeket a névvel összefűzve őrizd meg minden
esetben.
7. A nem tipikus lakhelymegjelöléseket is vedd figyelembe, pl.
településnév + házszám vagy, településnév + kerületszám.
7.1. Ha a lakhelyet követő sorban számszerű adat van, akkor az
a hely adatokhoz tartozik.
7.2. A kerületszám római számmal van jelölve, pl. VII, III, V.

Válaszodat a JSON adatmezőjébe helyezd ilyen formában (és kizá-
rólag a JSON-el válaszolj, szöveges összefoglalást vagy egyéb
hozzáfűzést ne tegyél):

[     HU_MNL_...|...: [
      {
  „tanu_1_nev”: „sor: [hipotézisek]”,
  „tanu_1_lakhely”: „sor: [hipotézisek]”,
  „tanu_2_nev”: „sor: [hipotézisek]”,
  „tanu_2_lakhely”: „sor: [hipotézisek]”
}],
      ...
]
```
[9]

A kimenettel kapcsolatos problémákat nem szüntette meg teljes mértékben, azonban ezzel a megfontolással kezdtünk bele a qwen3:235b-instruct[10] utasításkövető verziójának tesztelésébe.

9 Vö.: Az első prompt variációban elvárt kimenet: {‚tanu_1_nev': ‚adat', ‚tanu_1_lakhely': ‚adat', ‚tanu_2_nev': ‚adat', ‚tanu_2_lakhely': ‚adat'}

10 https://ollama.com/library/qwen3:235b-instruct (Utolsó letöltés 2026.05.08)

Nem jelentkezett kihagyás sem az ID-kat, sem a batch egyes példáit illetően. Jellemző hiba maradt azonban a címkék keverése, ilyenkor például tévesen a lakhelyhez került a név. Ennek áthidalására egy konfigurációs Modelfile-ban megadtuk a futtatáshoz lefoglalt GPU-k számát (160) és a kontextusablak nagyságát tokenben (4096), amely a futtatási környezetben ideális GPU-felhasználást eredményezett. Ezen kívül a hőmérsékletet 0,1-re állítva, egyszerűsített system prompttal („Answer immediately and directly. Do NOT use <thought> tags. Start your response with the final answer.") futtatva a modell jól működött a címkézési feladatra és nem produkálta a korábbi modellek jellemző hibáit. A tanúk neve, lakhelye mező 182 példáján 84%-ban százalékban jól teljesített. Hibás címkézés olyan esetekben fordult elő, amikor csak egy lakhely volt megjelölve, vagy az utca neve egy személynév.

További javulást sikerült elérni a promptban meghatározott kimeneti példák egyértelműsítésével. A tanúk neve, lakhelye mező esetében pontosabb eredmények születtek:

```
{
„tanu_1_nev": {
    „1. sor": [„hipotézis"],
    „2. sor": [„hipotézis", „hipotézis"]
  },
„tanu_1_lakhely": {
    „3. sor": [„hipotézis"],
    „4. sor": [„hipotézis"]
  },
„tanu_2_nev": {
    „5. sor": [„hipotézis"],
    „6. sor": [„hipotézis", „hipotézis"]
  },
„tanu_2_lakhely": {
    „7. sor": [„hipotézis"],
    „8. sor": [„hipotézis"]
  }
}
```

4. Következtetések

Fontos megjegyezni, hogy ez egy folyamatban lévő kutatási és fejlesztési projekt, jelen tanulmány pedig az eddigi eredményeket és tapasztalatokat foglalja össze. Elmondható, hogy a modellek teljesítménye mezőtípusonként eltérő, ezért a promptok módosítása és a feladat egyszerűsítése (pl. csak szegmentálás kérése) elengedhetetlen a pontosság növeléséhez.

Vannak mezők, amelyeken a jelenlegi beállításokkal már jó eredményt érünk el, de egyik modell sem képes megbízhatóan kezelni az összetettebb szerkezetű mezőket, és könnyen lehet, hogy nem találunk egységes modellalapú megoldást minden mezőre.

Végső soron sikeresnek bizonyult a megközelítés, amely a feladatot részekre bontotta és egy lépésben kevesebbet várt el az LLM-től. Ez azonban további fejlesztést igényel, mivel modellek használatát szabályalapú eljárásokkal kell ötvözni a fennmaradó feladatok ellátására.

Bár egyes mezőkön már most is kiválóan teljesítenek a nagy nyelvi modellek, az összetettebb struktúrák továbbra is kihívást jelentenek, és elképzelhető, hogy nem találunk univerzális megoldást minden mezőre; az eddig tesztek során nyert tapasztalatok – a think=False hatékonysága a „gondolkodás" visszaszorításában, a rendszerpromptok és in-context példák szerepe az ellentmondások csökkentésében, valamint a qwen3:235b-instruct relatív stabilitása – azt mutatják, hogy a gyakorlatban a legjobb eredmény a feladat finom részekre bontásából és a nyelvi modellek célzott prompt- és paraméter-hangolásával kombinált szabályalapú ellenőrzésekből adódik.

5. Függelék

5.1 Anyja/apja/elhunyt neve, foglalkozása, lakhelye

```
JSON:
{
„HU_MNL_PVL_XXXIII_0001_a_0001_c_0005_0268|1”: [
    {
      „1. sor”: [
        „Tóth Imréné született Zeszenezki Te-”,
        „Tóth Imréné született Zeszenczki Te-”,
        „Tóth Imréné született Leszenczki Te-”,
        „Tóth Imréné született Zeszenczki Te¬”
      ]
    },
    {
      „2. sor”: [
        „véz, férje lakásn”,
        „véz, férje lakása”,
        „néz, férje lakása”,
        „vér, férje lakása”,
        „vér, férjé lakása”
      ]
    }
  ]
}
```

5.2 Menyasszony/vőlegény adatai

```
JSON:
{

  „HU_MNL_PVL_XXXIII_0001_a_0001_b_0006_0042|2”: [
    {
      „1. sor”: [
        „Klein Regina”
      ]
    },
    {
      „2. sor”: [
        „izraeltita”,
        „izraeltíta”
      ]
    },
    {
      „3. sor”: [
        „szül. 1884. június 2.”
      ]
    },
    {
      „4. sor”: [
        „lak. Abony I”,
        „loh. Abony I”
      ]
    },
    {
      „5. sor”: [
        „nelei u. 121.”
      ]
    }
  ]
}
```

5.3 Tanúk neve és lakhelye

```
JSON:
{
  „HU_MNL_PVL_XXXIII_0001_a_0001_b_0006_0042|2”: [
    {
      „1. sor”: [
        „Kagrenki”,
        „Kagvenki”,
        „Kugremki”,
        „Kugrenki”
      ]
    },
```

```
    {
      „2. sor”: [
        „Ligost”,
        „Ligosát”,
        „Lijosb”,
        „Lijosh”,
        „Lispát”
      ]
    },
    {
      „3. sor”: [
        „Abony II”
      ]
    },
    {
      „4. sor”: [
        „Andrássy u.”,
        „Andróssy u.”
      ]
    },
    {
      „5. sor”: [
        „Schwarcz”
      ]
    },
    {
      „6. sor”: [
        „Mór”
      ]
    },
    {
      „7. sor”: [
        „Abony II”
      ]
    },
    {
      „8. sor”: [
        „Andrincey úr.”,
        „Androssy u.”,
        „Androssy úr.”,
        „Andrászág u.”,
        „Andrásány u.”
      ]
    },
    {
      „9. sor”: [
        „23.”
      ]
    }
  ]
}
```