Multidiszciplináris tudományok, 10. kötet. (2020) 2 sz. pp. 363-371 **https://doi.org/10.35925/j.multi.2020.2.40**

TERMÉSZETES NYELVŰ INTERFÉSZ ARCHITEKTÚRA E-TUTOR KÖRNYEZETBEN

Megyeri Balázs
hallgató, Miskolci Egyetem, Informatikai Intézet
3515 Miskolc, Miskolc-Egyetemváros, e-mail: megbalaa@gmail.com

Kovács László
egyetemi tanár, Miskolci Egyetem, Informatikai Intézet
3515 Miskolc, Miskolc-Egyetemváros, e-mail: kovacs@iit.uni-miskolc.hu

Absztrakt

Az ember-számítógép terület mindig is fontos részét képezte az informatikának. Már az első modern komputereknél is szempont volt, hogy a gép az ember számára könnyen kezelhető legyen. Igény született az ember-gép kapcsolat egyszerűsítésére és emberibbé tételére. Ezt az igényt kezdetben a grafikus interfészek elégítették ki, azonban a 21. századra az emberek lényegében összenőtek a technológiával, így szükségessé vált ezen kapcsolat természetes nyelvi elemekkel való kibővítése, így megalkotva az ember-gép kommunikáció eddigi legeffektívebb módszerét.

Kulcsszavak: *NLP, természetes nyelv feldolgozás, Google Dialogflow, mesterséges intelligencia, neurális hálózatok*

Abstract

The human-computer field has always been an important part of informatics. Even with the first modern computers, it was important for the machine to be easy to use for humans. There was a need to simplify and make the human-machine relationship more humane. This need was initially met by graphical interfaces, but by the 21st century, humans had essentially merged with technology, making it necessary to expand this connection with natural language elements, creating the most effective method of human-machine communication to date.

Keywords: *NLP, natural language processing, Google Dialogflow, artificial intelligence, neural networks*

1. Bevezetés

Az emberiség mindig igyekezett eszközeit emberivé és emberközelivé alakítani. Elsőként a fizikai szintű, manipuláció alapú interfészek alakultak ki. Későbbiekben megjelentek az absztraktabb grafikus vagy verbális kommunikációs csatornák is. Napjaink egyik újdonsága az intelligens ember-gép, ember-számítógép interfész kifejlődése. Ezen interfésznek egyik leglátványosabb megjelenései a chatbotok, melyek a természetes nyelvű párbeszéd megvalósítását célozzák meg. Nagyon sok intelligens, smart alkalmazásban fontos komponens a természetes nyelvű interfész (NLI), mely azonban jelenleg még a fejlődése középső szakaszában van.

A jelen cikk fő témája, a természetes nyelvű interfészek vagy ismertebb nevükön a chatbotok alkalmazási lehetőségeinek vizsgálata az e-tutor rendszerekben. Az e-tutor rendszerek az e-learning alapú alkalmazások azon típusa, amikor a diák és a tanító rendszer között egy adaptív, smart párbeszéd alakul

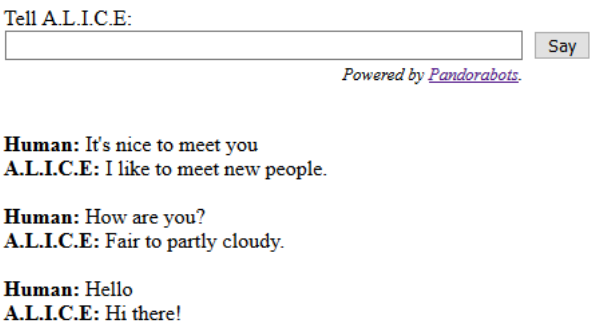
ki, mely alapvetően az NLI technológiára épít. Az intelligens diák – rendszer párbeszéd ugyan sokban hasonlít az általános chatbot kommunikációs modellre, de igen sok egyedi vonással is rendelkezik, melyekre e cikkben kívánunk rámutatni.

2. A chatbotok története

A természetes nyelven történő kommunikáció bár modern és újkori elképzelésnek hangzik, mégis már nagyon régóta foglalkoztatja a tudományt. A történelem során számos mérföldkő jelent meg ezen és az ehhez kapcsolódó szakterületeken, de általánosan elmondható, hogy a mesterséges intelligencia és az emberhez hasonlóan gondolkodó rendszerek alapjai a 2. világháború után, az 50-es években kezdtek el megjelenni az első számítógépekhez kötődően.

Első mérföldkőnek Alan Turing tekinthető, aki 1950-ben megjelent cikkében [2] felvetette a kérdését „Tudnak-e a gépek gondolkodni?”. Ez a felvetés már akkoriban is korszakalkotónak hatott, annak ellenére, hogy az akkori gépek még igencsak messze álltak a könnyen kezelhetőségtől és az emberi nyelvvel való kapcsolattól. Cikkében továbbá felvázolt egy eljárást is (Turing-teszt), mely szerint az intelligencia azzal mérhető, hogy egy gép mennyire képes emberszerűen kommunikálni. Sok tekintetben ezt a művet tekinthetjük a mesterséges intelligencia kutatás és a chatbotok bölcsőjének. [1]

1966-ban Joseph Weizenbaum elkészítette ELIZA nevű chatbotját, amivel a Turing-tesztet szerette volna teljesíteni. ELIZA egy pszichiátriával foglalkozó chatbot volt, aki kérdéseket tett fel és ezzel próbált segíteni a felhasználóknak megérteni saját érzéseiket, gondolataikat.[3] A teszt teljesítése sajnos nem sikerült, mivel túlságosan merev volt maga a rendszer. Előre betáplált válaszai voltak és szimpla kulcsszavak és kulcsmondatok segítségével próbálta kategorizálni a megadott bemenetet, viszont így is jelentős eredményeket tudott felmutatni, hiszen már voltak, akik elhitték róla, hogy ember, illetve megközelítésével alapvető elveket fektetett le a chatbotok fejlesztését illetően.[1]



1. ábra. *Beszélgetés A.L.I.C.E-al*

1995-ben Richard Wallace fejlesztett egy A.L.I.C.E. nevű online chatbotot, ami számos díj mellett elnyerte a Loebner-díjat is, mely a legemberszerűbb chatbotot díjazza. Maga a bot JAVA nyelven íródott és az AIML (Artificial Intelligence Markup Language) nevű XML sémát használja, melyben effektíven lehet megadni a szabályok, kérdések és válaszok halmazait, melyek közül a rendszer később választani fog [4]. Ennél az alkalmazásnál tehát már megjelent az igény egy strukturált kezelőfelületre, ahonnan könnyedén módosítható a bot tudásbázisa, illetve egy könnyen átlátható és a gép számára is gyorsan feldolgozható fájlra, melyben lementhetjük ezt az idővel egyre nagyobb méretű adatbázist. [1]

2010 környékétől jelentős ugrás következett be a chatalkalmazások fejlesztésében. A hatalmas techóriásoktól kezdve az éppen feltörekvő startup cégekig mindenki kezdte felismerni a chatbotokban rejlő

lehetőségeket. Ebben a fejlődésben jelentős szerepük volt az akkoriban egyre jelentősebb eredményeket elért neurális hálózatoknak. Míg korábban a 70-es 80-as évekig féltek és nem láttak jövőt ezekben a rendszerekben, addig a 2000-es évektől egyre jobban kezdtek megbarátkozni ezzel a már meglehetősen régen felfedezett, de mégis a korszak jellegéből adódóan újszerűen ható, biológiai eredetű technológiával.

3. A chatbotok működése

A történeti áttekintésünkben vázolt chatbotok működése meglehetősen egyszerű és túlságosan merev volt. Legtöbbjük egyszerű elágazásokon, reguláris kifejezéseken és statisztikai képleteken, megfigyeléseken alapult. Emiatt is nagyon könnyen össze lehetett zavarni őket, sok esetben teljesen eltértek a kontextustól, vagy szimplán nem értették, amit a felhasználó írt nekik.

Napjaink chatbotjai azonban számos új technikát alkalmaznak ezen problémák kiküszöbölése érdekében kezdve a modernebb statisztikai elemzésektől, a morfológiai vizsgálatokon át, egészen a neurális hálózatokig. Ezen megoldásokban a közös, hogy mindegyikük vagy valamilyen ember által alkotott struktúrát visz bele az elemzésbe (például nyelvi elemzés) vagy valamilyen randomitásra épít, de a lényeg, hogy szinte teljesen megváltak a statikus elemektől.

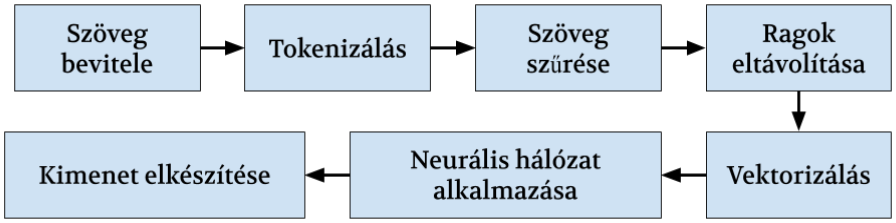
Az ezeket a chatbotokat működtető NLP algoritmusok mindegyike más és más, de mégis felfűzhető működésük egy általános folyamatra vagy algoritmusra, melynek bemenete az értelmezni kívánt szöveg lesz, kimenete pedig maga az értelmezés.

Az első ilyen lépés az magának a szövegnek a bevitele. Ez történhet írásban, hangok alapján hangfelismerés segítségével vagy kamerákon, képeken keresztül karakterfelismeréssel. Ezen lépés során szimplán a karakterek megfelelő formátumban történő beolvasására koncentrálunk.

Következő lépésben a már bevitt szöveget elkezdjük egységekre bontani, melyet tokenizálásnak hívunk. Ez az egység lehet szó, mondat vagy akár betű is feladatunktól függően. Maga az egységekre bontás jelentősen függ a bemenet nyelvétől, hiszen ilyenkor valamilyen karakter alapján bontjuk fel a szöveget (jellemzően vessző, pont vagy szóköz karakter alapján), ami erősen nyelv függő.

Ezután következik a már felbontott szöveg szűrése. Ilyenkor felesleges szavakat és ún. stopwordöket keresünk, melyek vagy nem bírnak jelentéssel az adott feladat esetén, vagy esetleg zavaróak lehetnek az algoritmus számára.

Következő lépésünk egy szintén részben opcionális teendő, az azonos gyökerű szavak felkutatása és a ragok eltávolítása. Számos nyelvben, köztük a magyarban vagy az angolban is a szavak rendelkezhetnek különböző módosítókkal, ragokkal, melyek kiegészíthetik az adott szó jelentését.



2. ábra. *Szövegfeldolgozás lépései*

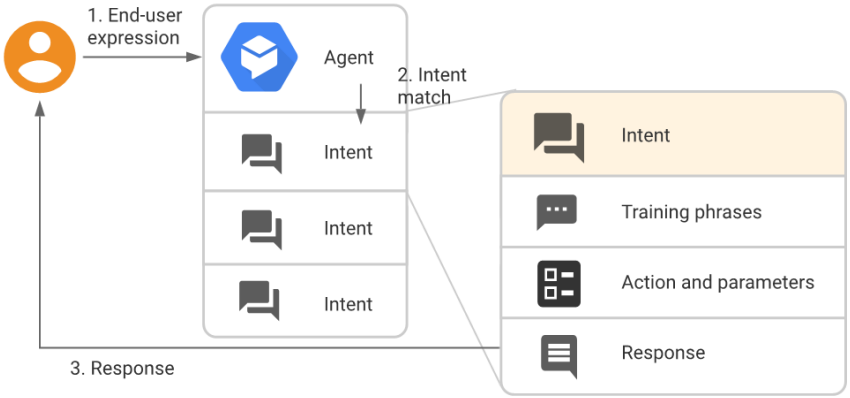
Ezen kiegészítések azonban nem mindig szükségesek számunkra, hiszen már az adott szó gyökeréből is következtethetünk annak szövegben betöltött szerepére, de bizonyos esetekben pont ezen ragokra van szükségünk, például, ha el szeretnénk dönteni, hogy az adott szöveg milyen időben íródott.

Utolsó előtti lépésünk a felbontott szövegünk vektorizálása és ezen vektorok neurális hálózatokkal történő feldolgozása lesz. Ebben a lépésben a már tiszta szövegünket a hálózat számára értelmezhető formába alakítjuk és végigfuttatjuk hálózatunkon. A feldolgozás történhet felügyelt tanulást alkalmazó módszerekkel, ilyenkor előre betanítjuk a hálózatot és ezek alapján fog mintákat keresni, vagy történhet nem felügyelt módszerrel, mely során az algoritmus magától fog keresni összefüggéseket a szövegben.

Végül algoritmusunk biztosít egy kimenetet, mely lehet egy diagram, egy konkrét válasz objektum, vagy csak szimplán valamilyen elemi struktúra. Ezt az eredményt ezután tesztelhetjük, analizálhatjuk és kiértékelhetjük, míg végül eljutunk a helyes kimenetig. Ezt a kimenetet aztán rendszertől függően elmenthetjük, elküldhetjük válaszban, vagy újra felhasználhatjuk algoritmusunk bemenetének.

4. Google Dialogflow keretrendszer

A Google Dialogflow rendszerében a természetes nyelvű interfészt ágensnek hívjuk. Ez egyfajta gyűjtőhelye lesz az általunk definiálandó céloknak és kontextusoknak. Ezt az ágensünket összekapcsolhatjuk REST API-n keresztül vagy a Google saját könyvtárai segítségével alkalmazásunkkal, illetve van lehetőség más nagyobb chatalkalmazásokkal való kooperációra is.



3. ábra. *Dialogflow ágensek működése [5]*

Belső működését tekintve minden ágens rendelkezik egy saját neurális hálózattal, illetve lehetőség van az egyes bemenetek szabály alapú algoritmusokkal való feldolgozására is. Ha az adott céljainkhoz sok példamondat vagy kifejezés áll rendelkezésünkre, akkor célszerűbb a neurális hálózat alapú algoritmusokat választanunk ágensünkhöz, míg ha kevés példa áll rendelkezésünkre, akkor a szabályalapú algoritmusok használata az előnyösebb. Természetesen lehetőség van hibrid megoldásokat is alkalmazni, melyek a célnak megfelelően fogják alkalmazni ezen algoritmusokat.

A Dialogflowban lévő entitások megfeleltethetőek a valós világ entitásainak és itt, mint intent-jeink paraméterei jelennek meg, melyeket a rendszer felismer és szelektálja őket típusonként. Ahhoz, hogy ezen entitások felismerésre kerüljenek entitás típusokat kell definiálnunk ágenseinkhez. Ezekhez a típusokhoz ezután példákat és szinonimákat kell biztosítanunk, melyeket, mint bemeneteket kezel majd a rendszer neurális hálója és így képes lesz kategorizálni az egyes entitásokat.

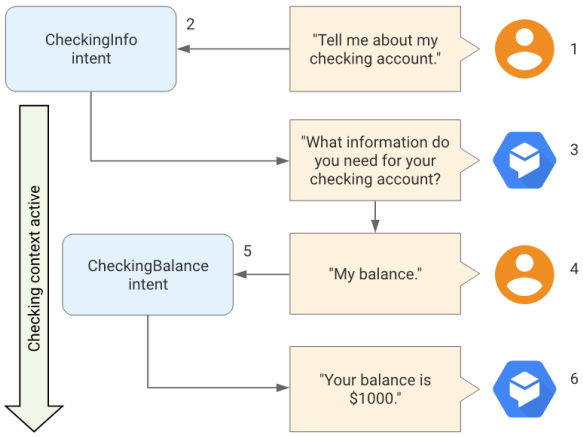
A beszélgetési célok a Dialogflow rendszer legfontosabb komponensei. Mint egyfajta kisebb célok csoportosítják a felhasználó kívánalmait és ezek felismerésével fogja tudni a rendszer, hogy éppen mit is kéne válaszolnia az adott bemenetre. Ezen célok együttese fogja alkotni magát a chatbot-ot és annak teljes beszélgetési mechanizmusát. A célok kezelése hasonló az entitásokéhoz. Itt is példákat kell

biztosítanunk a rendszernek, mely ezáltal megtanulja, hogy az egyes bemenetekre hogyan kell reagálnia. Régebbi rendszerekhez képest azonban a különbség, hogy itt a példák megadásával sokkal pontosabban tud különbséget tenni az algoritmus az egyes célok között, mely köszönhető részben a neurális hálónak, részben pedig számos NLP és statisztikai módszernek.

Az célok végül egy válaszobjektummal fognak visszatérni, melynek tartalmát a grafikus felületről kezelhetjük. A válasz lehet egyszerű szöveges vagy tetszőleges tartalmú JSON válasz. Szöveges esetben válaszlehetőségeket kell megadnunk és a rendszer az adott cél felismerésekor választ egyet közülük, míg a JSON esetében tetszőleges tartalommal tölthetjük fel, akár kontextus változókra és paraméterekre is hivatkozhatunk benne, melyek értékét később behelyettesíti a rendszer.

Akárcsak a való életben a természetes nyelvek esetén, így a Dialogflow-ban is létezik kontextus. Ezen kontextusokat a célokban állíthatjuk be és az aktuális session tárolja őket, mely a bottal való kapcsolatfelvétel esetén jön létre és saját egyedi azonosítóval rendelkezik.

Minden célnak van egy bemeneti és egy kimeneti kontextusa, melyekből akár több is lehet. A bemeneti kontextus egyfajta korlátozást fog adni a cél felismerésére, vagyis mindegyiknek aktívnak kell lennie ahhoz, hogy a cél felismerésre kerüljön, míg a kimeneti kontextus beállít az adott session-ben egy kontextust adott élettartammal, mely élettartam minden újabb cél felismerésével csökken. Ezt az élettartamot az API-n keresztül kezelhetjük, tehát adhatunk hozzá és el is vehetünk belőle.



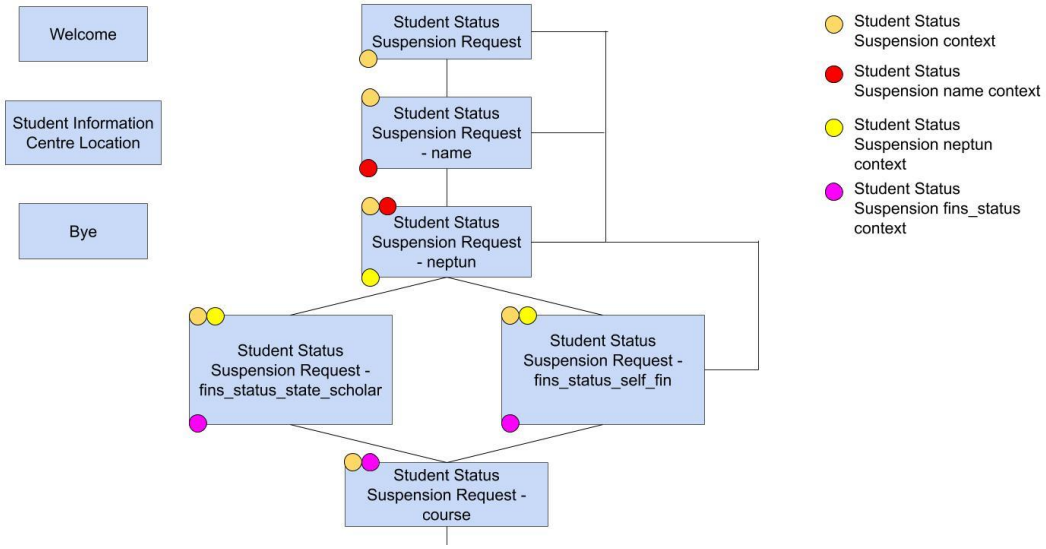
3. ábra. *Dialogflow kontextus [6]*

A kontextus az intentek felismerése mellett egyfajta memória szerepet is betölt, mivel tárolhatunk benne különböző paramétereket, melyeket a bemeneti szövegekből nyer ki a rendszer. Megadhatjuk, hogy milyen néven tárolja le ezeket paramétereket, kötelezőek-e, illetve alapértelmezet értéket is állíthatunk be hozzájuk. Ezen paraméterek az API válaszobjektumában automatikusan tárolásra és behelyettesítésre kerülnek a kontextus hierarchiának megfelelően, tehát a legfelsőbb, legátfogóbb kontextusból elérhetők az alsó kontextusok paraméterei.

5. Egy mintarendszer bemutatása

Mint minden alkalmazás esetében az első lépés a specifikáció készítés és a tervezés. A chatbot témája az oktatás és a hallgatói tevékenységek segítése. Maga a bot egy információs és kérvénykezelő rendszer NLP megfelelője, melyen keresztül információkat lehet lekérni az adott egyetemről, valamint kérvényeket lehet vele létrehozni egy egyszerű természetes nyelvű interfészen keresztül.

Ezután következik az egyes beszélgetési célok és kontextusok összegyűjtése, illetve rendszerezése. Itt hasznos lehet egy folyamatábra elkészítése, melyre bár még nincs hivatalos szabvány, de érdemes rajta felvenni a beszélgetési célok nevét, a beszélgetés folyamatát nyíllal vagy vonallal mutatni, illetve az egyes bemeneti és kimeneti kontextusokat és azok élettartamát is meg lehet rajta jelölni.



4. ábra. *A beszélgetési célok folyamatábrája (részlet)*

Az ábráról leolvashatóak az egyes célok és az esetenkénti kontextusaik. A célok bal felső sarkában a bemeneti, míg bal alsó sarkukban a kimeneti kontextus szerepel. Amelyik cél nem rendelkezik kontextussal az csak egy szimpla választ fog visszaadni, illetve aktiválásához nincs szükség egyéb kontextusra.

A célok és kontextusok után még lejjebb mehetünk a tervezésben és elkezdhetjük az egyes célokat példamondatokkal körülírni. A célunk itt az, hogy összegyűjtsük az összes olyan bemenetet, amit a felhasználók az adott céllal összekapcsolhatnak. Itt kell megválasztanunk a bot szövegének a stílusát, melynek illeszkednie kell a fő témához.

Ahhoz, hogy a bot betölthesse nyilvántartó rendszeri szerepét, szüksége van valamilyen memóriára is. Ezt Dialogflowban és a legtöbb hasonló rendszerben kontextusokon és entitásokon keresztül lehet megvalósítani. Felvehetünk különböző entitásokat, melyeket meg kell ismertetnünk a rendszerrel. Ez történhet a célokhoz hasonlóan példák segítségével, melyeket a neurális háló feldolgoz és megtanul, vagy használhatunk reguláris kifejezéseket és egyéb szabály alapú megoldásokat. Ezeket a felvett entitásokat ezután használhatjuk beszélgetési céljaink példáiban, ahol a rendszer megtanulja, hogy az adott helyen valamilyen változó entitás fog szerepelni és ennek értékét később ki is tudja szűrni nekünk.

A rendszer által kiszűrt entitások és azok értéke kontextusokban kerül tárolásra. Az egyes kontextusok között egyfajta hierarchia áll fent.

Utolsó teendőnk a chatbot tesztelése. Egy valós alkalmazás esetében nyilván szükséges a backend és frontend funkciók tesztelése, azonban itt számunkra elegendő az NLP komponens vizsgálata. Az alábbi táblázat az elvégzett tesztek témaköreit foglalja össze.

1. táblázat. *A bot felhasználói tesztelése*

A bot felhasználói tesztelése		
Teszt mondat	Felismert cél	Kiszűrt entitások
hi	Üdvözlés	-
where can i get information about student activities?	Hallgatói Központ helye	-
where can i find the Student Information Centre?	Hallgatói Központ helye	-
i want to suspend my student status	Passzív félév igénylése	-
im looking for room A/1 245	Terem keresés	A/1 245 (terem)
where can i find C2/113?	Terem keresés	C2/113 (terem)
where is Dr. Nagy Zoltán s office?	Oktatói iroda keresése	Dr. Nagy Zoltán (oktató)
How much is my tuition fee if i begun my studies in 2016.09.01?	Tandíj lekérdezése	2016.09.01 (dátum)

6. E-tutor-beli alkalmazás architektúrája

Az e-tutor rendszerek egyik fő sajátossága, hogy egy szűkebb témakörben történik a kommunikáció, melynek célja lehet

- ismeret átadása,
- ismeret ellenőrzés,
- ismeret átadás igénylése.

Az e-tutor környezet és alkalmazás sajátosságait figyelembe véve meghatározhatóak azon új egyedi architektúra elemek, melyek kiegészítik a standard chatbot architektúra elemeket. A következőkben ezen új elemeket foglaljuk össze.

Egyik lényegi tulajdonság, hogy a dialógus céljai (intent) ebben a környezetben hierarchikus felépítésűek. Például a „másodfokú egyenletek megoldása” az „egyenletek megoldása” témakör alatt, annak speciális eseteként jelenik meg. Ezen specializációs kapcsolatok kezelésére egy cél-hierarchia struktúra jelenik meg a háttér adatbázisban.

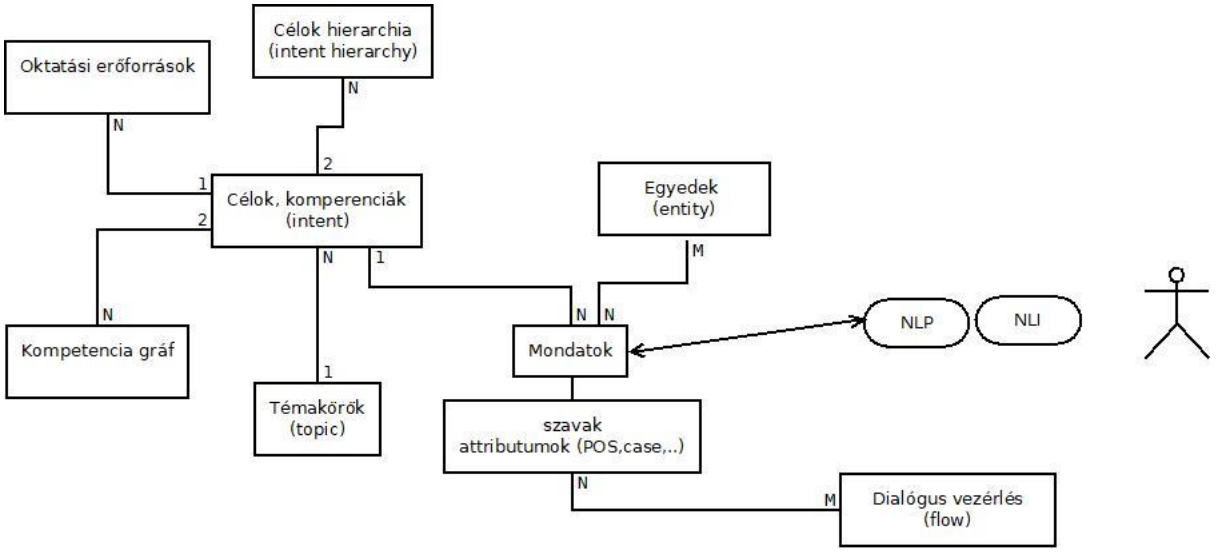
A célok specializációs hierarchiája mellett szükség van egy témakör (topic) szintű csoportosításra itt. Egy-egy témakör a szakterület egy-egy szeletét jelöli ki. Ilyen témakör lehet például az „algebra”, mely magába foglalja a „másodfokú egyenletek megoldása” témakört is. Mivel a témakör szerinti szűkítés egyik fontos elem a tanítási folyamat során, a háttér adatbázisnak tehát tartalmaznia kell egy ilyen témakör szerinti csoportlírást is.

A harmadik igényelt speciális komponens az e-tutor környezetben a célok függőségi kapcsolatainak a tárolása. A függőség itt egyfajta kompetencia szintű függőséget jelent, vagyis azt mutatja, hogy egy adott cél teljesítéséhez (például ahhoz, hogy meg tudjuk oldani a másodfokú egyenletet), milyen más célok (kompetenciák) teljesítése szükséges. Ez a függőségi gráf igen fontos szerepet tölt be többek között a dialógus folyam (flow) megvalósításánál is, hiszen az e-tutor rendszernek egy sikertelen hallgatói teljesítés esetén fel kell tárnia, hogy mely alap-kompetenciák hiányoznak, mely kompetenciákat kell

gyakorolni, erősíteni a hallgatónál. Emiatt a harmadik fő új adatbázis komponens a célok kompetencia gráfja.

Az egyes célok mögött ott kell állnia egy a kapcsolódó erőforrás elemeknek is, melyek különböző formátumú, az adott kompetenciához tartozó oktatási erőforrásokat jelölnek ki.

A felvázolt elemeket is figyelembe véve megadhatjuk a felépíthető a mintarendszer javasolt architektúráját. A 6.ábrában megjelenített struktúra összefoglalóan tartalmazza a javasolt NLI interfész kulcs elemeit és azok kapcsolatait.



5. ábra. *MLI motor architektúra e-tutor keretrendszerhez*

A javasolt architektúra több egyedi, új szoftvermodul megvalósítását igényli. A meglévő ChatBot keretrendszerek nem biztosítanak kellő rugalmasságot és funkció készletet, emiatt egy alapjaiban új rendszer kifejlesztését tartjuk célszerűnek.

7. Összefoglalás

Mint láthattuk a természetes nyelv feldolgozás és maguk a chatbotok is jelentős fejlődésen mentek keresztül. Napjainkra már nyugodtan mondhatjuk, hogy készen áll a szakterület a vállalati, vagy egyéb globálisabb használatra is, azonban még sok terület van, ahol fejlesztésre és optimalizálásra van szükség. Látható, hogy a mesterséges intelligencia kutatása jelentősen előre lendítette az NLP fejlődését, azonban azt is érzékeljük, hogy ebben a formában ez nem lesz elegendő a valóban gondolkodó interfészek készítéséhez. Még szükség van valamire, ami egyelőre csak az emberi agyban van jelen, több rétegű, tudat alatti működésekkel dolgozik, ötletekkel lát el és működése is komplexebb. Ennek kutatása és feltérképezése lehet az elkövetkezendő évtized fő feladata.

8. Köszönetnyilvánítás

A cikkben ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi

fejlesztése" projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Irodalom

[1] ZemČík, T.: *A brief history of chatbots*, DEStech Transactions on Computer Science and Engineering, 2019 **https://doi.org/10.12783/dtcse/aicae2019/31439**

[2] Turing, A. M.: *I. - Computing machinery and intelligence*, Mind 1950, 236(59):433-460. **https://doi.org/10.1093/mind/LIX.236.433**

[3] Weizenbaum, J.: *ELIZA - a computer program for the study of natural language communication between man and machine*, Communication 1966, 9(1):36-45. **https://doi.org/10.1145/365153.365168**

[4] Wallace R.S. The Anatomy of A.L.I.C.E.. In: Epstein R., Roberts G., Beber G. (eds) Parsing the Turing Test. Springer, Dordrecht, 2009. **https://doi.org/10.1007/978-1-4020-6710-5_13**

[5] https://cloud.google.com/dialogflow/docs/intents-overview (letöltés dátuma: 2020.04.08.)

[6] https://cloud.google.com/dialogflow/docs/contexts-overview (letöltés dátuma: 2020.04.16.)