

Magyar nyelvű időpont-egyeztető dialógusrendszer v2

Nagy Soma Bálint¹, Herdinai Viktor¹, Pálfi Gellért¹, Fegyő Tibor^{2,3},
Mihajlik Péter³, Farkas Richárd¹

¹ Szegedi Tudományegyetem, Informatikai Intézet

² SpeechTex Kft.

³ Budapesti Műszaki és Gazdaságtudományi Egyetem,

Távközlési és Médiainformatikai Tanszék

nagysomabalint@gmail.com, rfarkas@inf.u-szeged.hu,

tfegyo@speeche.com

Kivonat: A dialógusrendszerek fejlesztése napjaink fontos kutatási területévé nőtte ki magát. Munkánkban a korábban bemutatott időpont-egyeztetésre szolgáló chatbotunk felhasználói visszajelzésekből származó tapasztalatait tárgyaljuk, valamint részletezzük, hogy az így megtalált hibákat/hiányosságokat milyen módon javítottuk ki. A legfontosabb fejlesztésünk, hogy a dialógus állapotainak leírására egy teljes új reprezentációt vezetünk be, ami az időpont-egyeztetés, mint tárgyalási feladatot általánosan tudja támogatni. Célunk kezdetektől fogva a telefonos időpont-egyeztetés volt. Bemutatjuk chatbotunk beszéd-alapú interfészre való adaptálásának első lépéseit, egy kis mintás kiértékelésen keresztül.

1. Bevezetés

A chatbotok elképesztő popularitásra tettek szert az elmúlt évtizedben. Míg korábban elsősorban cégek fejlesztettek, egyszerű kulcsszókeresés-alapú rendszereket, az elmúlt néhány évben a számítógépes nyelvészetet is elérte a chatbot láz. Amennyiben a magyar üzleti chatbotok bonyolultabb és természetes emberi interakciókat akarnak kezelni, akkor szükségük van magyar nyelvi jelenségek kezelésére alkalmas algoritmusokra. Azért, hogy a jelenlegi chatbotok határait és továbbfejlesztési igényét megértsük, egy chatbot alkalmazás fejlesztési lépéseit mutattuk be Nagy és mtsai. (2021) cikkünkben.

Az alkalmazás üzleti/társadalmi célja, hogy a felhasználó magyar nyelven időpontot egyeztessen bármilyen környezetben. Médiaként elsősorban beszéd-alapú (telefonos) ügyfélszolgálatok mögötti dialógus rendszert képzelünk el, ezért csak folyó szöveges kommunikációra koncentrálnak (és például nem vizsgáljuk a chatbotok által gyakran használt kattintással listaelem választó lehetőségét).

Jelen cikkünkben bemutatjuk az időpont-egyeztető chatbotunk első verziójára kapott visszajelzéseket, és a visszajelzések alapján végzett fejlesztéseket. A jelen rendszert tekinthetjük kettes verziónak (v2). A chatbot forráskódja, illetve a kulcsfontosságú részét alkotó dátum- és időpontentitások felismerésére szolgáló könyvtár elérhető a <https://github.com/szegedai/hun-appointment-chatbot> és

<https://github.com/szegedai/hun-date-parser> oldalakon, illetve működése tesztelhető a publikusan elérhető az <https://www.inf.u-szeged.hu/algmi/chatbot/> oldalon.

2. Kapcsolódó munkák

Számos chatbotokkal és dialógus rendszerekkel foglalkozó nemzetközi publikáció érhető el, köztük néhány, amelyben időpontegyeztető rendszert mutatnak be a szerzők. A francia nyelvű időpont egyeztetést célul kitűző CLASSiC projekt keretében többféle megoldás született a problémára, köztük a részlegesen megfigyelhető Markov döntési folyamatokra épülő ún. “System 2” (Jurcicek és mtsai., 2010). A NASTIA rendszer (L. El Asri, 2014) olyan módon fejlesztette tovább a korábban publikált megvalósításokat, hogy a dialógus egyes, előre meghatározott állapotaiban megerősítéses tanulás segítségével választ a rendszer a rendelkezésre álló stratégiák közül. Az ilyen tanulórendszerek esetén szükséges jutalmat az egyes beszélgetések után a felhasználó által adott elégedettségi pont jelentette. A DINASTI (L. El Asri, 2014) publikációban a NASTIA rendszer alkotói közzétették a tanításra, valamint a kiértékelésére készített 1734 francia nyelvű dialógust tartalmazó korpuszt.

Az utóbbi időszakban a mélytanulás térnyerésével a dialógusrendszerek megvalósítására a szabályalapú rendszerek helyett a teljes egészében adatvezérelt, “end-to-end” tanított dialógusrendszerek kerültek az előtérbe. Eric és Manning (2017) a neurális dialógusmodellek feladatorientált dialógusrendszerek esetén való alkalmazási nehézségeit tárgyalja, és az ún. kulcs-érték kinyerő háló (key-value retrieval network) architektúrát javasolja a problémák megoldására. A szerzők által tárgyalt három scenárió egyike az időpontegyeztetés, a cikkben tárgyalt rendszer tanítása során használt időpontegyeztetési dialógusokat is tartalmazó angol nyelvű adathalmazt közzétették.

A magyar nyelvű chatbotokról szóló publikációk száma csekély. Kemény és Recski (2018) Tervezz Velem utazásszervező Facebook Messenger chatbotjának célja a BKK API lekérdezésével a felhasználók utazást érintő kérdéseit megválaszolni. Chatbotjuk képes volt a nyelvileg töredékes, rövid üzenetekből is pontos felhasználói szándék kinyerésére. A Lara könyvtár (Nagyfi, 2018) egy célzottan magyar nyelvű chatbotok fejlesztésére készült eszköz, felhasználói szándék meghatározását, entitások kinyerését teszi lehetővé, emellett pedig számos nyelvi feldolgozó eszközt tartalmaz magyar nyelvre.

Korábbi munkánkban (Nagy és mtsai., 2021) bemutattuk az időpont-egyeztető chatbotunk első verzióját, amely képes volt a felhasználó és a bot között több körben, a felhasználó szöveges üzeneteit felhasználva, nagyon egyszerű dialóguslogikát követve időpont-egyeztetést lefolytatni. Chatbotunk a gépi tanulásra alapuló, nyílt forráskódú Rasa keretrendszert használja (Bocklisch és mtsai., 2017), amit a spaCy-hez

készült magyar nyelvi feldolgozó eszközök, illetve CBOW szóbeágyazás (Orosz, 2019) segítségével alkalmaztunk a magyar nyelvre.

Korábbi cikkünkben bemutattuk továbbá a chatbotok fejlesztésére használható nyílt forráskódú keretrendszereket, valamint tárgyaltuk a chatbotunk szempontjából kritikus jelentőségű természetes szövegből dátum-, illetve időpontentitások kinyerésére használható eszközöket. Részleteztük az általunk fejlesztett és közzétett dátum-, időpont- és időintervallum-kinyerő program célját és működését. Jelen cikkünkben bemutatjuk az időpont-egyeztető chatbotunkon végzett fejlesztéseket.

3. Felhasználói felület a chatbot tesztelésre

Annak érdekében, hogy felhasználói visszajelzések segítségével javíthassunk a chatbotunkon, megértjük a valós emberi beszélgetés során elkövetett jellemző hibákat, a chatbothoz egy publikusan elérhető webes felhasználói felületet fejlesztettünk: <https://www.inf.u-szeged.hu/algmi/chatbot>. Külön figyelmet szenteltünk a visszajelzési lehetőségekre, és minden felhasználói interakciót tároltunk (anonim módon), hiszen a célunk az ember-gép diskurzusok mélyebb megértése volt.

Fontos, hogy minden beszélgetést eltároljunk, mivel ezek mutatják meg, hogy hol tudunk javítani a chatbot működésén, ehhez használtuk a Rasa TrackerStore nevű funkcióját. A TrackerStore lehetőséget nyújt egy adatbázis-szolgáltatás (például MongoDB, PostgreSQL, Redis) egyszerű integrálásához. Esetünkben a MongoDB NoSQL adatbázisra esett a választás. A dokumentumokban el van tárolva a beszélgetés azonosítója, a felhasználó üzenetei, a chatbot válaszai, a felismert intent és megannyi fontos adat. A dokumentumokat lehet szűrni valamennyi mezője alapján, illetve mivel JSON-szerű formátumban (BSON) vannak tárolva, egyszerűen lehet őket JSON fájlba exportálni, ezen fájlokból pedig könnyedén tudjuk rekonstruálni a felhasználó és bot közötti beszélgetést olvasható formátumban.

Lehetőséget teremtettünk a beszélgetéssel kapcsolatos azonnali visszajelzésre is. A szöveges visszajelzéseket a beszélgetés azonosítójával és időpontjával szintén egy MongoDB adatbázisban tároljuk, az azonosító alapján pedig rálinkeljük a szóban forgó beszélgetés logjára. Egy kérdőíves visszajelzést is feltettünk, de erre egyelőre még statisztikailag értelmezhetetlenül kevés kitöltés érkezett.

4 Visszajelzések az első verziós időpont-egyeztetőről

Az elkészült webes felületen 2021. szeptembere és novembere között 80 beszélgetés zajlott a chatbottal. Két fő alapos tesztelőnk volt (senior product managerek), rajtuk kívül körülbelül 15¹ programozó szakos egyetemi hallgató próbálta ki a chatbotot.

¹ Mivel a teljes anonimitásra törekedtünk, nem tárolunk cookiekat és semmilyen információt, amiből vissza lehetne következtetni, hogy melyik beszélgetést ki folytatta.

A chatbotról visszajelzést adó fent bemutatott felhasználó felület, illetve a bot forráskódjának Github oldalán kapott visszajelzésekből és beszélgetések átolvasásából a következő problémacsoportokat azonosítottuk.

4.1. Felhasználói szándék felismeréséhez köthető hibák

Chatbotunkban a felhasználói szándék felismerését egy transzformer-alapú, neurális háló architektúra, a DIET (Bunk és mtsai., 2020) végzi. Mivel mi a dátum és időentitások megállapítására a saját, szabályalapú entitáskinyerőnket használtuk, így csak a felhasználói szándék kinyerésére alkalmaztuk a DIET-et. A Rasa-ba a szöveg előfeldolgozására a magyar nyelvű spaCy-t integráltuk (Orosz, 2019).

A visszajelzésekből világossá vált, hogy a chatbot felhasználói szándék felismerésére szolgáló komponense nem eléggé robusztus, érzékeny a felhasználói üzenetekben történő kis változásokra is. Például többször megjelenő hiba a felhasználói üzenetekben megjelenő különféle írásjelek, speciális karakterek okozta felhasználói szándék tévesztés volt. A bot például az “Üdv” üzenetet helyesen osztályozta, azonban az “Üdv!” esetében hibásan állapította meg a felhasználó szándékát (amely mindkét példa esetén a bot kezdeti üdvözlése lett volna a felhasználó részéről).

Több lehetséges megoldás vizsgálata után végül amellet döntöttünk, hogy megvárjuk az új magyar spaCy pipeline elkészültét és nem kezdünk bele az előfeldolgozási lépések javításába. Gyors megoldásként egyszerűen eltávolítjuk a speciális karaktereket az előfeldolgozás során, ami a hibák egy részét orvosolja. Eddig ez nem okozott újabb problémákat (pedig a mondatvégi írásjel fontos lehetne a szándék azonosításában). Az NLU pipeline-ban, a kezdeti tokenizáló lépés elé (miután meggyőződünk arról, hogy az nem konfigurálható úgy, hogy a speciális karaktereket eltávolítsa) egy általunk implementált lépést helyeztünk el, amely eltávolítja a speciális karaktereket a szövegből. Ezt a modult a Rasa pipeline architektúrájával kompatibilisen kellett implementálnunk.

4.2. A dátum és időpont entitás kinyerésének hibái

A dátum és időpont entitások kinyerését a saját, erre a célra implementált könyvtár végzi. Ennek jellemzője, hogy szabályalapú és tiszta szöveges bemenetet feltételez. A felhasználói bemenetet 16 lehetséges komplex reguláris kifejezés illesztése után, számos heurisztika segítségével kíséreljük meg értelmezni, ennek köszönhetően az entitás kinyerő tapasztalataink alapján megbízhatóan működik

- számos formában megadott dátum és időpont,
- ékezetek, kis-nagybetű különbségek, minimális elírások, egyes gyakori rövidítések,
- számok szöveges formában történő kiírása,
- illetve hiányos dátum vagy időpont kifejezések esetén.

Mivel azonban a jelenleg használt módszer azt igényli, hogy minden felismert scenáriót előre manuálisan kezeljünk, így elkerülhetetlen, hogy a rendszer hibázzon, amikor egy olyan formában kifejezett dátum- vagy időentitást kap bemenetül, amely felismerését megelőzőleg nem implementáltuk. A közzétett chatbottal folytatott beszélgetéseket tekintve az ilyen esetek súlya nem túlnyomó, azonban a beszélgetésre való hatása miatt mindenképp kezelni kell ezeket a hiányosságokat. Erre a típusú hibára néhány példa:

- Találkozunk *két hét múlva*. (A könyvtárban nem volt megvalósítva az *n időperiódus múlva* forma értelmezése.)
- Ráérek *17-18h* között. (A könyvtár nem volt képes az SMS-nyelvben az óra kifejezésére használt *h* értelmezésére.)

A fenti és ehhez hasonló hibákat teljesen megszüntetni ugyan nem lehetséges, de a felismert szerkezetek bővítésével képesek voltunk számos korábban gyakran előforduló hiba orvoslására.

Szintén az entitáskinyerőnk számlájára írható hiba volt, hogy amikor ugyan az egyes dátum és időelemeket helyesen azonosítjuk a bemeneti szövegben, azonban a heurisztika, ami a végső kimenetet alkotja meg a felismert elemek ismeretében, hibás vagy hiányos. Erre példa:

- *Nyolckor* kezdődik a vacsora. (Mivel a szövegből explicit módon nem derül ki a napszak, így hibásan reggel 8 órát nyert ki a szövegből a program.)

Erre a konkrét hibára részleges megoldást jelentett, hogy lehetővé tettük az időpontok kinyerését a bemeneti szövegből már ismert, tágabb időtartam megadásával kiegészítve, a fenti példa esetén, ha előre meghatározzuk, hogy a kinyerni kívánt időpont esti, akkor a program helyesen 20:00-t nyer ki a szövegből. Ez tette lehetővé, hogy a bottal folytatott többkörös időpont egyeztetés során, ha a felhasználó és a bot már megállapodott egy tágabb időtartamban, akkor a felhasználó azt anélkül tudja tovább specifikálni, hogy teljesen pontosan meg kelljen határozni a kért, szűkebb intervallumot. Jelenleg a könyvtár rendelkezik olyan függvénnyel, ami paraméterül várja a tágabb intervallumot, illetve a szöveget, amelyből a tágabb intervallum ismeretében kívánunk időpontot kinyerni. Például, ha a felhasználó és a bot korábban már megegyezett abban, hogy nem az aktuális, hanem a soron következő hétre keresnek időpontot, akkor megadható a következő hét, mint tágabb intervallum, így a könyvtár a *péntek* szöveghez az e hét péntek helyett a jövő pénteket rendeli hozzá.

4.3. A dialógus limitált állapotmenedzsmentje okozta hibák

Egy dialógus állapota alatt a felhasználóval folytatott üzenetváltás korábbi üzeneteiből kinyert, felhasználható formában eltárolt, a beszélgetés folyamatát később befolyásolni képes információkat értjük. Első verziós chatbotunk az ún. szlot kitöltési (slot filling) dialógus menedzsment keretben volt implementálva, amelyet a Rasa - és minden egyéb chatbot fejlesztő keretrendszer - megvalósít. Első verziós chatbotunk a dialógus

irányítása során két szlot kitöltésre koncentrált: legyen egy olyan dátum (nap), amiben megegyeznek a felhasználóval a találkozót illetően, illetve a találkozó konkrét időpontja a napon belül. Ezt a két információt szöveges formában (ISO dátum formátum) tároltuk, amiket a bot a felhasználóval folytatott beszélgetés során olvasni-írni tudott.

Több felhasználói beszélgetésben megfigyelhető, hogy az előzetes feltételezésünkkel ellentétben két szlot nem elegendő a beszélgetés állapotának reprezentálásához. Gyakran fordult elő, hogy a felhasználó visszautalt egy korábban említett lehetséges dátumra, ha az éppen tárgyalt dátum nem volt megfelelő. Két szlittal ilyen sok információ eltárolása nem lehetséges, további szöveges szlotok felvétele átláthatatlanná tette volna a chatbothoz írt vezérlést. Erre a problémára megoldást az 5. fejezetben tárgyalt új dialógus állapotrepresentáció implementálása jelentett.

4.4. Felhasználói élménnyel kapcsolatos problémák

A beszélgetések átolvasásából szembetűnő volt, hogy a chatbot elmulasztotta a kezdeményezést, a beszélgetést a felhasználónak kellett kezdeményeznie, a bot nem adott semmilyen segítséget a felhasználónak azzal kapcsolatban, hogy milyen típusú probléma megoldásában képes asszisztálni. Az új verzióban a bot küldi az első üzenetet, bemutatkozik, elmondja, hogy milyen célt szolgál és konkrét kérdést tesz fel a felhasználónak.

Korábban az asszisztensnél magánál lehetett időpontot foglalni, a felhasználó arról érdeklődhetett az asszisztensnél, hogy ő, azaz a bot mikor ér rá. A természetesség és a hitelesség növelése érdekében a jelenlegi verzióban a bot egy képzeletbeli karakter, Főnök úr személyi asszisztense, így a felhasználó nem a bot szabad időpontjairól, hanem Főnök úr naptárával kapcsolatban érdeklődik.

Megjegyezzük, hogy erre a módosításra, nem csak a felhasználói élmény növelése, hanem a 2021. áprilisában megjelent EU rendelet (EB, 2021) miatt is szükség volt, hiszen az kimondja, hogy “Bizonyos MI-rendszerek, amelyek rendeltetése a természetes személyekkel való interakció ... a természetes személyeket figyelmeztetni kell, hogy MI-rendszerrel lépnek kapcsolatba”. Tehát megtévesztő lehetett a korábbi verzióknk, mert esetleg valaki azt hihette, hogy emberrel, és nem géppel beszél.

5. Új dialógus állapotrepresentáció

A legnagyobb limitációt az jelentett, hogy az állapotrepresentációt két darab, szöveg típusú szlittal valósítottuk a chatbotunk első verziójában. A Rasa keretrendszerben a szlotok kulcs-érték párok, a beszélgetés folyamán a bot memóriájaként szolgálnak. A szlotok száma és típusa előre meghatározott. A két szlot feladata a dátum, illetve az adott napon belül kölcsönösen megfelelő időpont eltárolása volt.

Azt, hogy ez milyen esetekben jelent limitációt, a következő annotált beszélgetés mutatja be:

Felhasználó:	Időpontot szeretnék jövő héten	<i>(“jövő hét” entitás felismerése, két jövő heti szabad nap ajánlása.)</i>
Chatbot:	A legközelebbi két nap amikor a Főnök úr ráér a kért időszakban jövő hét szerdán és jövő hét csütörtökön lesz.	
Felhasználó:	Esetleg csütörtökön?	<i>(“csütörtök” entitás felismerése, adott hét csütörtöki napjaként értelmezi a chatbot)</i>

A példában a szöveggörnyezetből kiderül, hogy a felhasználó csütörtöki időpont kérésekor valószínűleg továbbra is a jövő hétre keres időpontot, azonban, mivel ez az információ már nem része a bot memóriájának, így a bot képtelen rájönni, hogy jövő hét csütörtökről van szó.

A probléma igazi gyökere az, hogy az időpont-egyeztetési feladat fél úton helyezkedik el a szlot kitöltési feladat és a tárgyalási célú dialógusok között (El Asri és mtsai., 2014). Egyrészt időpont szlotokat akarunk kitölteni a felhasználóval, de az nem a klasszikus módon történik, mint mondjuk egy éttermi rendelő chatbotnál, hanem meg kell egyezni a mindkét fél számára alkalmas időpontban. A szlotok számának növelésével nem tudtuk volna a probléma igazi gyökerét orvosolni.

Ahhoz, hogy a komplexebb időpont-egyeztetési dialógusokat is meg tudjuk valósítani, újra kellett gondolnunk a dialógus állapotának reprezentációját, azt, hogy milyen adattípus választása lenne célszerű a bot memóriájaként való használatra, mi lenne az a forma, amelyben minden releváns korábban elhangzott információ tömören, strukturáltan tárolható, a későbbiekben kellően absztrakt módon lekérdezhető. Ez a probléma annak a vizsgálatával egészült ki, hogy milyen technikai korlátai vannak egy tetszőlegesen komplex adattípus szlotban való tárolására a Rasa részéről. A korábbi választás célszerűnek tűnt, mivel az általunk használt Rasa keretrendszer csupán Python által szeriazálható képes információkat tud szlotban tárolni, kizárva ezzel a nem automatikusan szeriazálható típusok (például az általunk is használt datetime, illetve DateTimeRange objektumok) triviális használatát.

Az új dialógusállapot-reprezentációt úgy alkottuk meg, hogy a következő szempontoknak feleljen meg:

1. Alkalmas legyen a bot (előre ismert) és a felhasználó (a dialógus folyamán kifejtett) szabad időpontjainak tárolására közös idővonalon, olyan módon, hogy ezek átfedései egyszerűen lekérdezhetőek legyenek
2. Tárolható legyen a korábban már kizárt (sikertelen végkifejlettel tárgyalt) időtartamok listája
3. Amennyiben a bot és a felhasználó több körös egyeztetést követően egy rövidebb konszenzuális időtartamra jut, ám azt az egyik fél végül elutasítja, a botnak legyen emléke az elutasított időpont előtti körben megegyezett időtartamról.

Az első pont szükségességét az indokolja, hogy szükség lehet a felhasználó által korábban említett lehetséges, azonban a beszélgetés során nem kifejtett időpont későbbi felidézésére. Például:

Felhasználó: **Időpontot szeretnék kedden vagy szerdán** (*“kedd, szerda” entitások elmentése, mint a felhasználó számára lehetséges elfogadható időintervallumok.*)
Chatbot: **Főnök úr kedden délután 2 és 4 óra között ér rá.**

A fenti beszélgetés esetén, noha a bot a kedd tárgyalásába kezd bele a felhasználóval, érdemes eltárolni, hogy a felhasználó egy másik lehetséges napot is megjelölt, ha a keddi napon végül nem sikerül kölcsönösen elfogadható időpontot találni.

A második pontot az indokolja, hogy nem szeretnénk a felhasználónak kétszer ugyanazt az időpontot ajánlani, ha korábban már elutasította a felajánlott időpontot.

A harmadik pont szükségességét a következő szcenárió illusztrálja:

Felhasználó: **Időpontot szeretnék jövő héten** (*“jövő hét” entitás felismerése, két jövő heti szabad nap ajánlása.*)
Chatbot: **Melyik nap lenne alkalmas?**
Felhasználó: **Kedd** (*“jövő hét” ismeretében a “jövő kedd” megjegyzése*)
Chatbot: **Főnök úr ráér jövő hét kedden délután 2 és 4 óra között.**
Felhasználó: **Esetleg 3 órakor?** (*“délután 3 óra” entitás felismerése*)
Chatbot: **Főnök úr ráér jövő hét kedden délután 3 órakor. Beírom a naptárjába a találkozót.**
Felhasználó: **Lehetne inkább csütörtökön?**

Az aktuális üzenetben észlelt időintervallumot a korábbi üzenetekben megtalált időintervallumok fényében szeretnénk értelmezni, tehát, ha tudjuk, hogy a legutóbbi üzenetben tárgyalt intervallum a *[jövő kedd 14:00; jövő kedd 16:00]*, akkor az aktuális üzenet *délután 3 óráját* a korábbi intervallum a fényében *jövő kedd 15:00*-ként szeretnénk értelmezni. Azonban, ha egy olyan entitást nyerünk ki, mint ebben az esetben a *csütörtök*, amit a korábbi intervallumok a *jövő kedd 14:00*, *[jövő kedd 14:00; jövő kedd 16:00]* és a *jövő kedd* fényében nem értelmes, addig szeretnénk visszalépkedni, amíg olyan intervallummal találkozunk, aminek tekintetében már értelmezhető a *csütörtök*, ez esetben ez a beszélgetés kezdetén említett *jövő hét* (ha nincs ilyen, akkor természetesen a teljes jövőn, korlátozás nélkül értelmezzük az aktuális dátum- és időentitást).

A feltételeinket kielégítő adatszerkezet tetszőleges számú idővonalból áll. Jelenleg három, a bot számára alkalmas, a felhasználó számára alkalmas, illetve a már kizárt

időintervallumokat tartalmazza. Ezeket a beszélgetés során bejövő információk alapján frissítettük. A három idővonal már jóval komplexebb dialógusok lebonyolítását teszi lehetővé, és ha szükséges, a jövőben újabb idővonalakkal bővíthetjük a reprezentációt (például ‘nem a legjobb, de ha kell megoldom’).

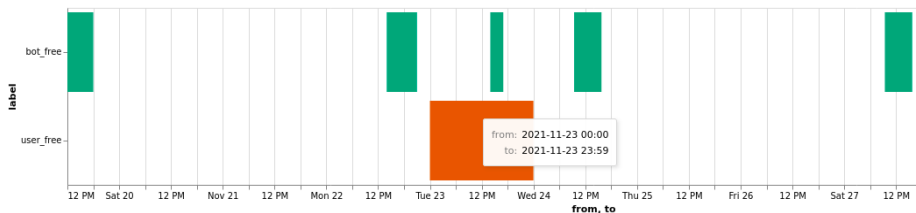
A dialógus stratégiánk célja, hogy az éppen tárgyalt időpontot “tölcseyszerűen”, egy tágabb időintervallumból a szűkebb felé terelje, de megteremtve annak a lehetőségét, hogy visszalépjünk egy fentebbi szintre, amennyiben szükséges.

A technikai akadályokon úgy léptünk át, hogy a saját adatszerkezetünkhöz megírjuk a szerializáló és deszerializáló eljárásokat, és a Rasa ezt így már képes kezelni, mint egyetlen speciális szlot.

Az új adatszerkezet előnye volt továbbá, hogy lehetővé tette, hogy az osztály felett, annak részeként az adattagokra vonatkozó komplex lekérdezéseket, módosításokat implementáljunk, így a dialógusvezérlés során megvalósított kódot letisztultá, olvashatóvá tettük.

Az időpont-egyeztetési folyamat követéséhez, a jelentkező hibák megértéséhez az idővonalakat, valamint az aktuálisan tárgyalt időintervallumhoz tartozó “tölcse” vizualizáló komponenst is készítettünk. Ennek célja az volt, hogy a beszélgetés minden pillanatában látható legyen, hogy az adott pontig milyen információ áll a bot rendelkezésére, milyen emlékek alapján határozza meg a választát. Például:

Felhasználó: **jövő kedden ráér?**
Chatbot: **Főnök úr ráér a kért időszakon belül, délután 2 és este 9 között.
Mikor lenne jó pontosan?**



1. ábra: A vizualizációs komponens kimenete.

6. Adaptáció audio-interfészre

Célunk a kezdetektől az volt, hogy az asszisztensünk hangalapon használható legyen. Ez volt az oka, hogy nem vizsgáltuk a lehetőségét olyan multimédiás opciók használatának, amelyek más szöveges chatbotokban gyakoriak, például gombok használata.

Ahhoz, hogy a chatbotunk hangalapon használható legyen, elkezdtük vizsgálni a magyar beszédleírató (speech-to-text, STT) chatbotunk pontosságára gyakorolt

hatását. Elkezdődött a SpeechTex Kft. beszédleiratózó rendszere (Mihajlik és mtsai, 2021) és az időpont-egyeztető chatbotunk technológiai integrációja. Az előbb említett publikációhoz képest abban különbözik a dialógusrendszerünkben alkalmazott beszédfelismerő megoldás, hogy a nyelvmodelljét a (Tarján és mtsai, 2016) szerinti közéleti, televíziós adatokkal tanított nyelvmodellel interpoláltuk.

Kezdeti lépésként arra voltunk kíváncsiak, hogyan reagál chatbotunk szándékfelismerési, valamint a dátumintervallum-felismerő komponense arra a potenciális zajra, amelyet az jelent, hogy nem ember által gépelt, hanem automatikusan leiratozott szöveg képzí a bemenetet. E két modul kiértékelése azért lényeges, mert ennek a két komponensnek a pontos működése kritikus a beszélgetés helyes alakulása szempontjából, bármelyik teljesítménye is romlik szignifikánsan, az a felhasználók beszélgetésméenyére közvetlenül rossz hatással lesz. A kis mintás kiértékelési kísérletünkben a 4. fejezetben bemutatott valós beszélgetésekből válogattunk 70 darab felhasználói közlést, és azt 6 fő (mind férfi, egy 41 éves, a többiek 21-24 évesek) felolvasta. A felolvasást számítógépes mikrofonnal végeztük el, mert a hangalapú időpont-egyeztetőnk első verziója demózási célokkal nem, csak mobil eszközön futó webes felületen keresztül lesz elérhető.

6.1. Szándékfelismerés kiértékelése

A szándékfelismerő komponens tanítása (*felhasználói szándék címke, felhasználói szándékot jelző mondat*) párok alapján történik. Kiértékeléséhez ilyen párokat gyűjtöttünk. A mondatok forrása a botot tesztelő felhasználók múltbeli üzenetei, az üzenetekhez tartozó címkéket többségi szavazással határoztuk meg.

A kiértékelést elvégeztük az eredeti mondatokon, valamint a beszédleiratózó modellt futtatva a 6 darab felolvasott változaton. A 6 darab felhasználói szándékot (osztályt) összesen 50 darab tesztmondattal kiértékelve az alábbi táblázat tartalmazza a pontosságot. A STT modell által leiratozott mondatok esetében a beszélőkre átlagolt pontosságot és szórást tüntetjük fel.

	Szándékfelismerés pontossága
Szöveg	77,0
Felolvasott majd automatán leiratozott	72,4 ± 4,3

1. táblázat. Szándékfelismerés kiértékelése.

6.2. Dátumfelismerő kiértékelése

A dátumfelismerő modul, lévén, hogy reguláris kifejezésekre alapul nem rendelkezik jó hibatoleranciával a bemeneti szövegben található zajra nézve, ezért különösen fontos

volt kiértékelni, hogy milyen hibára számíthatunk az STT modellünk által leiratozott szövegeken. Ehhez egy 20 dátumra vagy időre utaló kifejezésből álló listát gyűjtöttünk. A kezdeti listát úgy állítottuk össze, hogy a dátumfelismerő által ismert és kezelni képes eseteket tartalmazzon (azaz itt 100% a szövegből mért pontosság). Ezt követően a lista mondatait 6 olvasó olvasta fel, amit a STT modell segítségével leiratoztunk, majd az így képzett listákon is elvégeztük a kiértékelést. Az olvasónként átlagolt pontosság $90,84 \pm 5,4\%$ volt.

6.3. Beszédleiratozó típushibái

Ahogy az a fenti kiértékelési metrikák is alátámasztják, az általános beszédleiratozó is keveset hibázik. Néhány hiba:

Eredeti szöveg	Felolvasott majd leírt	Hiba
<i>ma vagy holnapután</i>	<i>ma a vagy holnapután</i>	hosszú <i>maaa</i>
<i>az túl korai nekem</i>	<i>hát az túl korai nekem</i>	extra <i>hát</i> (spontán beszéd)
<i>a főnökkel szeretnék beszélni</i>	<i>a főnöke szeretnék beszélni</i>	nyelvi modell hibája
<i>jövő hét</i>	<i>jövő 7</i>	speciális domain

2. táblázat. Beszédleiratozás hibái.

Ezeket a hibákat egy speciális nyelvi modellel, vagy akár a dátumfelismerőnk kiegészítésével is javítani lehet a jövőben.

7. Összegzés

Munkánk során a felhasználói visszajelzésekre, és a használat során megfigyelt gyakori problémákra reagálva több aspektusból is újra gondoltuk a korábban fejlesztett időpont-egyeztető chatbotunkat. Tapasztalataink alapján a dialógus új állapotrepresentációja gördülékenyebb, természetesebb dialógusokat eredményez. A továbbfejlesztett bot közzétételével, felhasználói visszajelések alapján kívánjuk megállapítani, hogy ez valóban beigazolódik-e. A chatbotot az <https://www.inf.u-szeged.hu/algmi/chatbot/> oldalon tesszük elérhetővé, a kipróbálóknak lehetősége van a hibák jelentésén túl egy kérdőív kitöltésére, ami a felhasználói élményükkel kapcsolatos kérdéseket tartalmaz.

Jövőbeli munkánk fókuszában a chatbot hangalapú verziójának elérhetővé tétele, és spontán beszéd leiratozása által keletkező új kihívások kezelése áll.

Köszönetnyilvánítás

A publikációban szereplő kutatást az Innovációs és Technológiai Minisztérium és a Nemzeti Kutatási, Fejlesztési és Innovációs Hivatal támogatta a Mesterséges Intelligencia Nemzeti Laboratórium keretében.

Hivatkozások

- T. Bocklisch, J. Faulkner, N. Pawlowski, A. Nichol: Rasa: Open Source Language Understanding and Dialogue Management. In Proceedings of NIPS workshop on Conversational AI (2017)
- T. Bunk, D. Varshneya, V. Vlasov, and A. Nichol: DIET: Lightweight Language Understanding for Dialogue Systems. arxiv.org/abs/2004.09936 (2020)
- F. Jurcicek, S. Keizer, F. Mairesse, K. Yu: D5. 4: Proof-of-concept CLASSIC Appointment Scheduling system (“System 2”). (2010)
- L. El Asri, R. Lemonnier, R. Laroche, O. Pietquin, H. Khouzaimi: NASTIA: Negotiating Appointment Setting Interface. In Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC’14 (2014)
- L. El Asri, R. Lemonnier, R. Laroche, O. Pietquin: DINASTI: Dialogues with a Negotiating Appointment Setting Interface. Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC’14 (2014)
- M. Eric, C. D. Manning: Key-Value Retrieval Networks for Task-Oriented Dialogue (2017)Európai Bizottság: A mesterséges intelligenciára vonatkozó harmonizált szabályok (mesterséges intelligenciáról szóló jogszabály) megállapításáról és egyes uniós jogalkotási aktusok módosításáról. AZ EURÓPAI PARLAMENT ÉS A TANÁCS RENDELETE. Brüsszel, 2021.4.21.
- Kemény Boldizsár, Recski Gábor: Természetes nyelvi interfész menetrend- és utazástervező szolgáltatásokhoz. In XIV. Magyar Számítógépes Nyelvészeti konferencia, (2018)
- Mihajlik Péter, Balog András, Tarján Balázs, Fegyó Tibor: End-to-end és hibrid mélyneuronháló alapú gépi leiratozás magyar nyelvű telefonos ügyfélszolgálati beszélgetésekre, In: XVII. Magyar Számítógépes Nyelvészeti Konferencia, pp. 139-145. (2021)
- Nagyfi R: Lara. <https://github.com/sedthh/lara-hungarian-nlp>. (2018)
- Orosz Gy.: Hungarian models for spaCy. <https://github.com/spacy-hu/spacy-hungarian-models>. (2019)
- Tarján Balázs, Varga Ádám, Tobler Zoltán, Szaszák György, Fegyó Tibor, Bordás Csaba, Mihajlik Péter: Magyar nyelvű, élő közéleti- és hírműsorok gépi feliratozása, In: XII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2016), pp. 89-99. (2016)