RAPORTARE ȘTIINȚIFICĂ

Proiect complex ReTeRom. Proiect component CoBiLiRo

Activitatea 3.4 - ***Proiectare de aplicații de exploatare a corpusului bimodal și a tehnologiilor de prelucrări textuale și voce, create în proiectele P2, P3, P4***

Faza de predare: noiembrie 2020

Autori: Pistol Ionuț, Scutelnicu Andrei, Mihaela Onofrei, Daniela Gîfu, Șerban Boghiu

1. Rezumatul etapei

În această etapă a proiectului complex ReTeRom consorțiul și-a propus să consolideze și apoi să exploateze rezultatele acumulate în primii doi ani, cu obiectivele: existența unui Portal pregătit a primi și prelucra resurse bilingve românești, dezvoltarea în continuare a unei colecții de resurse care să corespundă formatului agreat de consorțiu, perfecționarea lanțurilor de prelucrări lingvistice și sonore, atât asupra componentelor textuale cât și vocale ale resurselor bimodale, care să permită alinieri între componentele vocale și textuale, recunoașterea cu minimum de erori a vocii, generarea expresivă a vocii și antamarea de aplicații bazate pe aceste tehnologii.

A treia etapă (2020) a proiectului CoBiLiRO prevede completarea inventarului de resurse disponibile pe portal cât și valorificarea lor atât în cadrul platformei (statistici și instrumente integrate) precum și în afara platformei, propunând o serie de proiecte ce utilizează tehnologiile dezvoltate în cadrul proiectelor partenere. Similar celorlalte etape, este prevăzută și o activitate de diseminare, atât la evenimente științifice cât și în mass-media. De interes special pentru platforma dezvoltată este respectarea drepturilor de autor și a anonimizării solicitate pentru contribuitorii de resurse pe platformă.

2. Rezumatul activității

Activitatea 3.4 are ca obiectiv conceperea unor proiecte ce valorifică resursele colectate de platforma CoBiLiRo, utilizând tehnologiile dezvoltate în celelalte proiecte componente. Tehnologiile vizate în special sunt cele descrise în rapoartele 3.7 (Definitivarea, testarea, validarea și împachetarea într-o soluție „*ready-to-use*” a platformei integrate și configurabile de prelucrare a textelor în limba română.), 3.10 (Îmbunătăţirea soluţiei de filtrare şi aliniere a transcrierilor aproximative cu semnalul de vorbire), 3.15 (Dezvoltarea unei noi tehnologii pentru adaptarea vocii sintetice la stilul și expresivitatea unui nou vorbitor) și 3.17 (Integrare tehnologie nouă și demonstrare în realizarea interfețelor om-mașină pentru sinteza text-vorbire). Scopul acestui efort este de a demonstra potențialul tehnologiilor dezvoltate și de a mări vizibilitatea proiectului în special după finalizarea acestuia. O parte din proiectele descrise au depășit deja faza de proiect, fiind în stadii relativ avansate de implementare.

3. Descrierea științifică și tehnică

3.1 Scurtă descriere a tehnologiilor relevante

În secțiunile ce urmează sunt descrise o serie de proiecte ce au ca obiectiv valorificarea resurselor și instrumentelor produse în proiectele partenere, o sumară descriere a lor fiind dată imediat mai jos. O descriere a Platformei CoBiLiRo menționată poate fi consultată în rapoartele activităților A1.3 și A2.1.

În cadrul proiectului TEPROLIN a fost creată o platformă ce efectuează o serie de prelucrări asupra resurselor text (vezi rapoartele A1.5, A1.6 și A2.8). Serviciile oferite de acea platformă sunt incluse și pe serverul CoBiLiRo ca etapă automată de procesare a fișierelor text ce sunt adăugate la colecție.

Proiectul TADARAV a avut ca obiectiv principal dezvoltarea unei soluții de aliniere automată a unei resurse text-voce (vezi rapoartele A1.13 și A2.11). Aplicația rezultată a fost folosită pentru alinierea resurselor de pe platforma CoBiLiRO.

SINTERO dezvoltă o serie de instrumente pentru sinteza vocală, valorificând resursele CoBiLiRO cu adnotările produse de TEPROLIN și TADARAV (vezi rapoartele A2.15 și A2.18).

3.2 Aplicația 1: **Suport pentru învățarea limbii române - *PD builder***

Un dicționar de pronunție pentru cuvintele unei limbi poate constitui un suport semnificativ pentru cei care doresc să învețe acea limbă. Pe lângă dicționare, dedicate explicit acestui scop, cum ar fi Forvo[1], recent și Google include pronunția cuvintelor atât la căutarea specifică a lor cât și în sistemul de traducere automată.

Colecția de resurse colectate pe platforma CoBiLiRO are potențialul să atingă o dimensiune suficientă pentru a include pronunția marii majorități a cuvintelor din limba română, de aici a plecat ideea de a dezvolta o tehnologie ce construiește automat un dicționar de pronunții. Spre deosebire de dicționarele de pronunție existente, ce sunt construite prin înregistrarea sau generarea unui corespondent audio pentru fiecare intrare dintr-un dicționar existente, *PD builder* propune un proces de construire automată a unei astfel de resurse.

[1] https://forvo.com

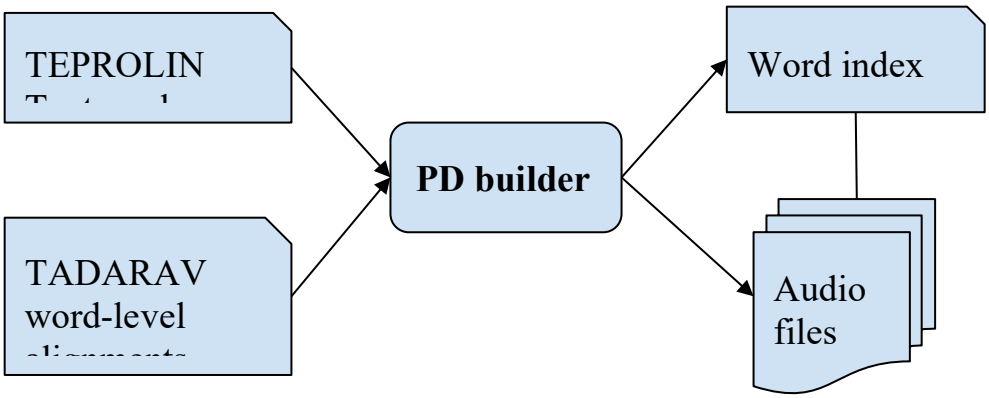

Figura 1: Arhitectura aplicației PD Builder

Aplicația propusă, numită *PD builder* (constructor de dicționare de pronunție), utilizează adnotările produse de TEPROLIN și alinierile produse de TADARAV, marcaje disponibile pe majoritatea resurselor existente pe platforma CoBiLiRo.

Primul pas în construirea dicționarului este indexarea intrărilor. Ca intrare considerăm atât forme de bază cât și eventualele forme flexionate disponibile. Corespondența între cuvântul de bază și formele flexionate se face pe baza lemei identificate de TEPROLIN. În cazul în care o intrare apare de mai multe ori în colecție, toate aparițiile sunt indexate. Acest lucru permite atât descoperirea unei pronunții standard (proces descris mai jos) cât și referirea la marcajele disponibile pe documentul sursă. De exemplu, un utilizator poate prefera ascultarea unei pronunții în contextul unei fraze/paragraf, fragment care poate fi recuperat din documentele sursă.

Pentru pronunția unui cuvânt, în momentul adăugării unei intrări sau referințe în index aplicația extrage din fișierul audio corespunzător sursei zona aliniată cu acea intrare conform marcajelor TADARAV. În cazul în care există mai multe pronunții pentru un cuvânt sunt extrase și indexate toate aceste variante. Un mecanism propus pentru această aplicație, posibil cu suportul tehnologiilor SINTERO, este construirea sau identificarea unei pronunții standard pentru acel cuvânt. Formele de undă corespunzătoare pronunției aceluiași cuvânt ar trebui sa fie similare, o medie a lor ar putea corespunde fie unei înregistrări disponibile, fie s-ar putea folosi pentru generarea unei înregistrări noi ce va fi asociată cu acel cuvânt. O metodă de clusterizare ar putea fi folosită pe setul de înregistrări, în cazul în care media este diferită semnificativ față de pronunțiile disponibile, pentru a descoperi eventuale alternative de pronunție. Astfel, pentru un cuvânt aplicația ar putea semnala și oferi două sau mai multe alternative de pronunție.

PD builder este momentan în dezvoltare, estimăm că un prototip va fi gata în prima parte a anului viitor.

3.3 Aplicația 2: **Analiza corpusurilor bimodale**

Soluțiile de aliniere automată a textului cu înregistrarea audio corespunzătoare (componentă a unui corpus bimodal voce-text) sunt descrise în raportul A1.11 [5]. Erorile în alinierile produse de aceste tehnologii au, în general, una din următoarele cauze:

- diferențe între transcrierea text și conținutul înregistrării;
- calitatea scăzută a înregistrării;
- erori ale aplicației de aliniere.

Se remarcă faptul că primele două surse de erori sunt independente de soluția de aliniere și influențează semnificativ calitatea potențială a alinierii produse precum și calitatea corpusului bimodal. Această idee de proiect își propune dezvoltarea unui sistem capabil să evalueze calitatea unui corpus bimodal voce-text din perspectiva unei alinieri automate sau manuale. Un astfel de sistem ar fi util atât dezvoltatorilor și utilizatorilor tehnologiilor de aliniere automată cât și celor care colectează și utilizează resurse bimodale în general. O parte din posibilele erori semnalate ar putea fi corectate, lucru care ar mări semnificativ calitatea alinierii automate dar și a resursei bimodale în general. O mare parte din soluțiile propuse mai jos necesită disponibilitatea cel puțin a unei metode de aliniere automată capabilă să ofere și scoruri de încredere pentru alinierile produse.

Diferențele între transcrierea text și conținutul înregistrării pot fi de mai multe tipuri:

- *Fragmente adiționale prezente în text sau în înregistrare*. Descoperirea automată a acestei situații poate fi făcută cu ajutorul unor soluții de aliniere automată ce ar putea să descopere zone din text sau audio care nu conțin alinieri precise. Eliminarea unor fragmente variabile din zonele aproximativ echivalente din cele două fișiere și retestarea alinierii automate ar putea descoperi o îmbunătățire a încrederii în alinierile descoperite, în special în zona fragmentului extras. Cele mai frecvente situații în care apare o astfel de eroare sunt cele în care fie componenta text, fie componenta audio, include pasaje adiționale la începutul sau sfârșitul fișierului, situație care facilitează descoperirea fragmentelor ce cauzează probleme. Dacă dimensiunile comparative în durată dintre cele două componente sunt evidente (de exemplu 120 minute durata audio și 500 de cuvinte fișierul text), prezența unor fragmente adiționale poate fi confirmată și fără utilizarea unui aliniator automat.
- *Transcrieri problematice*. Există situații prezente în text ce sunt dificil de transpus în audio, în special la o citire care nu are ca obiectiv transpunerea identică a conținutului textual. De exemplu, prescurtările, abrevierile și acronimele pot fi citite des în extenso, lucru care face dificilă alinierea ulterioară a celor două semnale. Numerele și datele calendaristice sunt alte exemple de situații similare. Descoperirea automată a acestor potențiale erori poate fi făcută de multe ori analizând doar fișierul text, dar dacă dispunem și de o aliniere produsă automat putem indica cu precizie zona echivalentă din audio unde eroarea este prezentă.
- *Particularități ale pronunției și prozodiei vorbitorului pe fișierul audio*. Cum aliniatoarele automate se bazează pe recunoașterea fonemelor și a granițelor (pauzelor) dintre ele, un vorbitor care nu are o dicție suficient de bună poate face aceste elemente mai dificil de descoperit. Aceste situații nu pot fi considerate însă de natură a diminua calitatea înregistrării, ele putând fi rezolvate în mare parte fără a schimba înregistrarea propriu-zisă

ci doar modalitatea în care aliniatorul o tratează. O antrenare/adaptare a parametrilor aliniatorului pentru o anumită persoană poate fi făcută în cele mai multe cazuri cu succes. Rămâne însă dificultatea descoperirii acestor situații, descoperire ce poate fi făcută automat prin observarea unor erori comune în alinierea anumitor cuvinte sau fragmente. Dacă într-o resursă aliniată automat apare un fragment aliniat greșit în mai multe locații, cel mai probabil acest lucru indică o particularitate a vorbitorului care ar necesita adaptarea aliniatorului.

Calitatea scăzută a înregistrării poate fi descoperită automat prin parametri precum: rata de eșantionare (*sample rate*), dimensiunea compresiei (*bit rate*), calitatea compresiei (*bit depth*). Ei pot fi detectați automat și pot indica o calitate generală a înregistrării. Pentru determinarea unor zone specifice în care calitatea semnalului audio e scăzută (zgomot de fundal, semnal slab), se poate apela din nou la un aliniator automat care ar indica o zonă în care alinierea e problematică fără a exista potențialul unor diferențe între text și audio (durate estimate aproximativ egale, textul nu conține situații problematice, text similar aliniat corect în altă parte a resursei). Analiza formei de undă a semnalului audio poate indica prezența unor voci adiționale ce perturbă calitatea înregistrării.

Acest proiect este în prezent în stadiu incipient, studii preliminare sunt făcute pe resursele disponibile pe platforma CoBiLiRo.

3.4 Aplicația 3: ***I listen to my speaking agent reading book fragments as I walk by***

Procesarea limbajului natural și transformarea textului în vorbire sunt componente esențiale ale aplicațiilor moderne, în special, ale celor de tipul interacțiune om - dispozitiv.

Aplicația are la bază o colecție de texte care abundă în entități geografice, marcate XML explicit, textele fiind însoțite de metadate care descriu minimum: autorul și titlul cărții, anul de apariție și editura. Instalată pe un dispozitiv mobil, ea va semnala proximitatea telefonului față de locațiile menționate în texte și va citi acele fragmente care includ mențiunile respective [3]. În felul acesta, o plimbare printr-un mare oraș se poate transforma într-o călătorie literară. Identificarea entităților reprezintă un prim pas în procesul de analiză, întrucât, astfel se obțin informații esențiale din textul analizat. Extragerea acestor informații a fost rezolvată (într-un proiect pilot realizat cu studenții Facultăți de Informatică de la UAIC) cu tehnici specifice extragerii de informații (*information extraction*) din texte, dar poate fi realizată și cu tehnologii incluse în setul de prelucrări pus la dispoziție de proiectul TEPROLIN.

Aplicația se adresează persoanelor care ar dori să primească sugestii de lecturi și doresc să afle lucruri noi despre locurile pe care le vizitează. Ea își propune să îmbine într-un mod plăcut literatura și tehnologia.

Provocări:

- Realizarea adnotărilor.

Pe decursul dezvoltării aplicației, adnotările au fost modificate de mai multe ori pentru a adăuga date noi necesare.

o Comunicarea între limbaje.

Modulul creat în Java și cel creat în Flutter comunică printr-un canal asincron și a necesară realizarea unui procedeu prin care să se asigure corectitudinea datelor schimbate între aceste module.

Tehnici, resurse și tehnologii utilizate:

Un prim pas în realizarea proiectului propus este adnotarea XML a unui document, pentru marcarea entităților ce desemnează locuri, instituții, monumente istorice etc., alături de crearea metadatelor care specifică informații referitoare la cărțile în care apar aceste entități, cu o marcare cât mai corectă și completă a acestora.

Chestiunea identificării şi a clasificării entităţilor cu nume, în implementarea experimentală realizată cu studenți, s-a făcut prin învățarea unor şabloane, capabile apoi să identifice și să clasifice chiar și entități care nu apar în textele adnotate. Metodele de clasificare a entităţilor sunt capabile să le asigneze acestora trei categorii: persoane, organizaţii şi teritorii.

Primele soluţii aduse pentru problema recunoașterii entităților cu nume (REN) s-au bazat pe aplicarea unor şabloane, reguli sau automate finite, în general create manual [4]. Această abordare presupunea expertiză umană pentru elaborarea sabloanelor, iar pentru că şabloanele create manual nu puteau acoperi toate cazurile de entităţi prezente în corpusuri mai mari, sistemele ulterioare au încercat să înveţe automat aceste şabloane din corpusuri adnotate, folosind diverse tipuri de reguli, transductoare sau automate finite. Cele mai recente studii şi aplicaţii în domeniul REN se bazează pe metode statistice, vectoriale și neuronale de învăţare automată.

Arhitectura proiectului conține 2 componente principale:

1.Aplicația Mobile.

Aceasta conține alte două submodule:

- ❏ Modulul principal realizat în Java, care adresează cereri către serverul Python;
- ❏ Modulul secundar realizat în Flutter, care se ocupă de sintetizarea text-to-speech.

2.Serverul Python.

Acest modul prelucrează textele adnotate și întoarce datele obținute în format JSON.

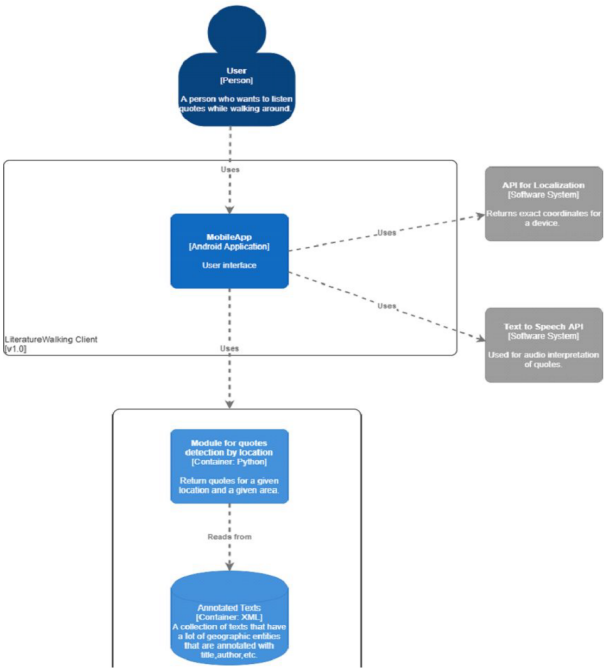

Fig.2 Arhitectura aplicației

Denumirea resursei: Flutter[2] Text to speech (flutter_tts)
Tipul resursei: Librărie opensource a limbajului Flutter folosită pentru transformarea textului în vorbire.
Scopul utilizării: Transformarea fragmentelor de text primite de la server în vorbire și redarea acestora
Producător: Tundra Labs
Descrierea resursei:

Flutter_tts este o librărie open source folosită pentru convertirea textului în vorbire. Această librărie poate fi folosită atât pe IOS, Android, cât și pentru dezvoltarea aplicațiilor web. Această librărie oferă diferite funcționalități, depinzând, însă, de platforma de pe care este folosită.

Cel mai simplu mod de utilizare ar fi apelarea directă a funcției *speak(text)*, textul care dorim să fie redat de un agent virtual. Acest text va fi transformat în vorbire în limba engleză, care este limba implicită în aplicația pilot. Pentru a schimba limba se poate folosi metoda *setLanguage()* care primește la intrare limba pe care dorim să o folosim. Această metodă trebuie apelată înainte de redarea textului. Pentru a opri redarea se poate folosi metoda *stop()*.

3.5 Aplicația 4: **Sistem de sinteză text-vorbire (TTS) și clonarea vocii în limba română cu metoda învățării prin transfer**

Modelul implementat la UAIC-FII presupune un modul de sinteză din text în spectogramă mel (sintetizator) și unul de generare a vorbirii din spectograma mel (vocoder). Ca sintetizator s-a

[2] https://pub.dev/packages/flutter_tts

ales Tacotron 2[3], antrenat inițial pe setul de date LJ Speech, iar ca vocoder - WaveGlow[4], antrenat inițial pentru limba engleză. S-au ales parametri audio identici pentru ambele modele, corespunzători cu parametrii LJ Speech.

Seturile de date în limba română au fost preprocesate, iar ambele modele au fost antrenate pe acestea. Sintetizatorul a fost antrenat pe un set de date aflat pe platforma RETEROM, corespunzător unui singur vorbitor, în timp ce vocoderul a fost antrenat pe un set de date cu mai mulți vorbitori. Din literatura de specialitate, pentru limba română au fost identificate implementări de sisteme de sinteză vocală, dar nu și de clonare a vocii, necesară, spre exemplu, pentru cei care și-au pierdut capacitatea de a vorbi.

Un alt scenariu pentru care clonarea ar fi utilă este învățământul de la distanță, unde tehnologia poate fi folosită pentru reproducerea vocii personalităților din istoria recentă și de azi.

Clonarea vocii, aici, reprezintă un caz particular de TTS în care un text este sintetizat în vocea unui vorbitor necunoscut folosindu-se un număr limitat de înregistrări audio ale acestuia acompaniate de transcrierea lor. Sistemul TTS poate fi conceptualizat ca un sistem de clonare a vocii unei persoane, însă sarcinile diferă prin accentul pus pe identitatea vorbitorului: în cazul TTS general, interesul este de a genera o voce cu caracteristici naturale cu cât mai puține greșeli de sinteză, în timp ce sarcina de clonare este de a genera vocea unei persoane specifice necunoscute, folosind cât mai puține mostre de vorbire ale acesteia. Au fost impuse următoarele restricții: (1) sistemul TTS general va fi antrenat pe maxim o oră de vorbire; (2) clonarea vocii se va face pe mai puțin de douăzeci de minute de vorbire; (3) sinteza vorbirii se va face în timp real; (4) totalul costurilor de procesare nu va depăși 300$; (5) soluția va fi un model *end-to-end*.

Validarea datelor

Pentru a valida setul de date, a fost folosit protocolul de selecție a datelor în scopul obținerii tehnologiilor TTS (*Dataset · mozilla/TTS Wiki · GitHub*). În urma verificării a 50 de fișiere audio, a fost găsită o singură greșeală de transcriere. În cazul folosirii metodei de selectare a unității, greșeala ar fi putut fi devastatoare pentru sistem, însă folosirea rețelelor neurale aduce avantajul robusteții față de erori, atât timp cât ele se anulează statistic una pe alta (greșelile nu sunt părtinitoare, cum ar fi transcrierea consistentă a fonemului „ț” prin litera „c”).

Preprocesarea datelor

Întrucât modelul Tacotron 2 a fost antrenat anterior pe setul de date LJ Speech, rata de eșantionate trebuie modificată la 22.050 Hz. Modalitatea *corectă* de modificare a ratei de eșantionare, pentru reproducerea rezultatelor este:

```
import librosa
```

[3] https://pytorch.org/hub/nvidia_deeplearningexamples_tacotron2/

[4] https://github.com/NVIDIA/waveglow

```
y, sr = librosa.load(wavFilePathFrom)

sf.write(wavFilePath, yt, 22050, format='WAV', endian='LITTLE',
subtype='PCM_16')
```

Următorul pas a fost ștergerea tăcerii de la începutul și finalul înregistrărilor audio. Timpul dinaintea vorbirii din corpusul antrenat surprinde vorbitorul inspirând auzibil pentru a se pregăti să citească textul, ceea ce nu a permis ștergerea cu ușurință a acestor segmente prin metoda eliminării segmentelor sub un anumit nivel de zgomot. Aici, de vreme ce fonemele au fost aliniate, tăcerea inițială și cea finală au fost aliniate manual, simplificând mult procesul de curățare audio.

Sintetizator text-spectogramă mel

Sintetizatorul Tacotron 2 folosit s-a dovedit eficient în procesul de aliniere, folosind un mecanism de atenție care grăbește procesul, permițând modelului să fie mult mai adaptabil. S-a folosit implementarea publicată de Nvidia (*GitHub - NVIDIA/tacotron2*), bazată pe modele preantrenate în limba engleză pentru sintetizator, cât și pentru două vocodere (WaveGlow și WaveNet), toate componentele bucurându-se de încorporarea tehnologiilor Apex (*Apex 0.1.0 documentation*). Acestea permit antrenarea rețelelor neurale folosind precizie mixtă, combinând viteza de calculare a reprezentării cu virgulă mobilă pe 16 biți cu precizia reprezentării cu virgulă mobilă pe 32 biți. Google Cloud (*Cloud Computing Services, Google Cloud*) oferă un an de gratuitate a serviciilor și 300$ pentru a experimenta diversele produse oferite, incluzând posibilitatea de a folosi o mașină virtuală care să beneficieze de procesare pe GPU. Aici, a fost folosit GPU Tesla T4 datorită faptului că prezintă *tensor cores*, ceea ce permite utilizarea tehnologiilor Apex, cât și a prețului modic în comparație cu alte procesoare ce presupun tehnologii similare (e.g. Tesla V100, de șase ori mai scump pentru același timp de rulare).

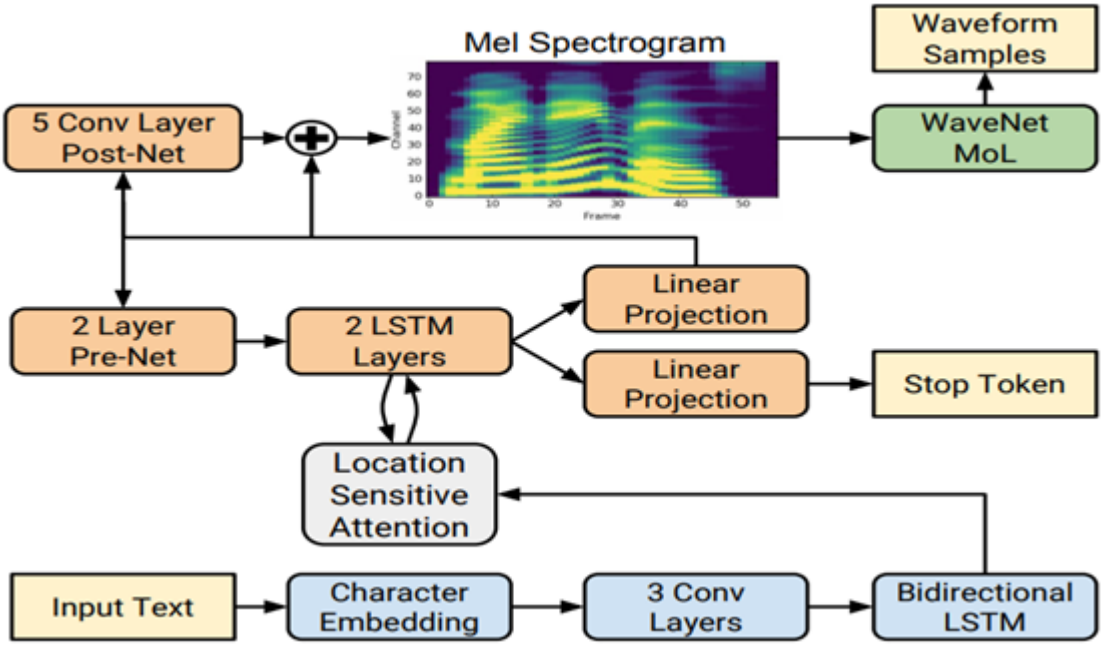

Fig. 3 Arhitectura modelului (după Tacotron 2)

Arhitectura modelului se bazează pe o rețea recurentă, care prezice spectrograme mel din text. Tacotron 2 este antrenat prin folosirea de perechi text-spectrogramă mel. *Spectrogramele mel* sunt un tip de spectrogramă obținute prin aplicarea unei transformări non-liniare axei de frecvență în cadrul transformării STFT (*Short-Iime Fourier Transform*) a formelor de undă, fiind inspirată de urechea umană, filtrând anumite frecvențe pentru a permite o reprezentare mai redusă, dar au ca efect exagerarea frecvențele joase, critice pentru inteligibilitatea vorbirii, cât și reducerea frecvențelor înalte, care sunt dominate de consoane fricative (z, s, f) și care, în general, nu sunt necesare pentru modelarea vorbirii. Detalii de implementare a arhitecturii sunt prezentate în repozitoriul Nvidia.

3.6 Aplicația 5: **Asistent inteligent al ședințelor online**

Întrucât în momentul de față pandemia Covid a obligat populația globului la migrarea majorității discuțiilor, ședințelor, întâlnirilor în mediul online, propunem realizarea unui instrument sau a unei colecții de API-uri, care să se poată integra aplicațiilor de conferințe online. Scopul acestora ar fi extragerea de informații și generarea de rapoarte din discuțiile purtate și din mesajele schimbate pe parcursul interacțiunilor online. Astfel, se poate imagina:

- extragerea automată a *chat*-ului;
- realizarea unui proces ce transformă convorbirea audio în text (*speech to text*), astfel încât toată conversația să fie transcrisă;
- realizarea unui rezumat al întâlnirii.

Având aceste câteva tehnologii implementate, se pot imagina funcționalități ale interfeței de conferință din mediul virtual, care ar permite participanților la dialog:

- extragerea de informații punctuale, doar pe anumite segmente din conferință (de exemplu pe durata apăsării unui buton) ori doar din intervențiile unor anumiți vorbitori, ceea ce ar elimina necesitatea ca utilizatorii să-și ia notițe în timpul conferinței, ceea ce le abate atenția de la discuții, putând duce chiar la omiterea unor informații esențiale;
- generarea automată a proceselor verbale, care ar trebui doar editate post-conferință;
- interogarea minutei ori a procesului verbal generat, pe bază de cuvinte cheie;
- căutarea în înregistrarea sonoră, a unor secvențe în care s-a menționat un anumit cuvânt cheie sau o anumită entitate, concept etc., cu reproducerea contextului vocal de apariție a lor.

3.7. Aplicația 6: **TRACKING ASSISTANT – Asistent inteligent pentru identificarea traseelor efectuate în decursul zilei**

Tracking Assistant este o aplicație Android simplă și intuitivă, care își propune să asiste persoanele care suferă de boala Alzheimer în a-și aminti traseul pe care l-au efectuat în timpul zilei printr-o interfață ușor de folosit, în limbaj natural. Arhitectura aplicației este prezentată mai jos.

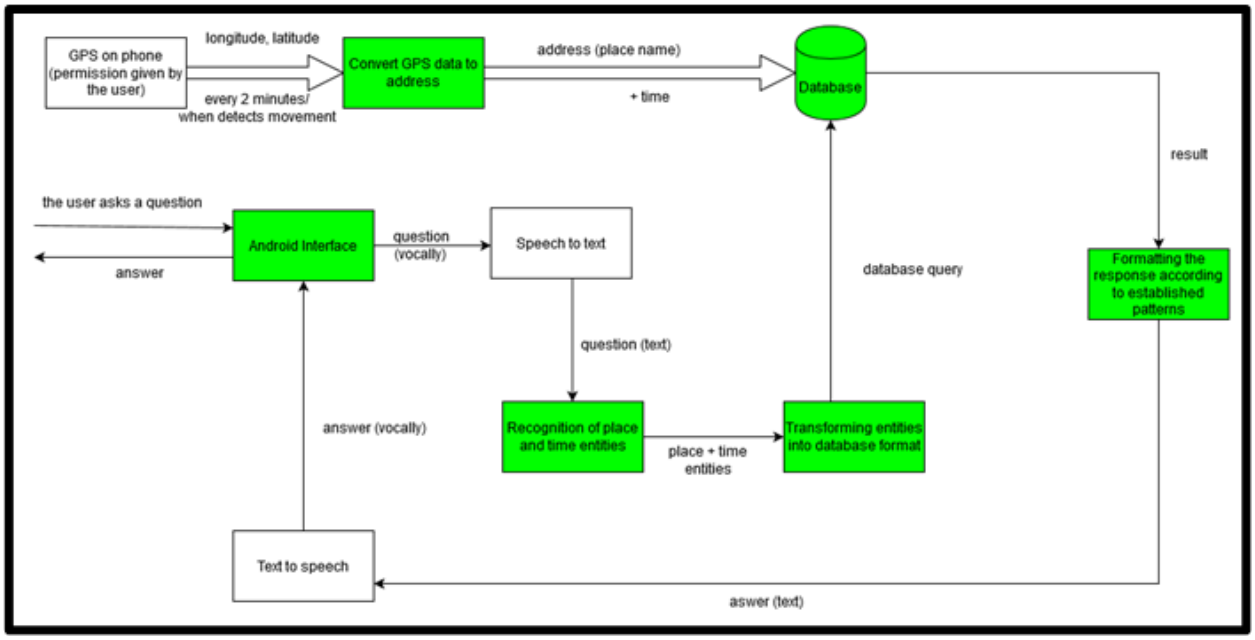

Fig. 4 Arhitectura aplicației Tracking Assistant

Primul modul se preocupă cu identificarea locației și stocarea sa în baza de date. După oferirea acordului asupra preluării datelor privind locația, aplicația accesează modulul GPS o dată la fiecare 2 minute sau atunci când este detectată mișcare [6]. Locațiile sunt stocate într-o bază de date împreună atât cu coordonatele GPS. Dacă locația are etichetă pe Google Maps (așa cum este,

de exemplu, cazul pentru unele școli sau universități, magazine, biserici etc.), aceasta este asociată în baza de date cu coordonatele corespunzătoare. Dacă nu, dar se observă că este o locație frecventată des, utilizatorul este întrebat dacă dorește să îi asocieze o etichetă (de ex. parc, poștă, magazinul din colțul străzii etc.). În caz contrar, este atribuită o etichetă bazată pe numele străzii celei mai apropiate, după Google Maps (de ex. "în apropiere de strada Nuferilor"). Această conversie a fost utilizată pentru a minimiza timpul de răspuns, deoarece în întrebările pe care le pune utilizatorul privind traseul nu sunt folosite coordonate GPS, ci nume de locații (sau nume aproximative). De asemenea sunt salvate coordonatele temporale pentru fiecare locație pentru un interval de 3 zile, interval care poate fi ajustat din setările aplicației. Deoarece volum de date care va fi stocat pentru 3 zile este destul de mare, fiecare locație prestabilită va denumi, de fapt, un areal (interval pentru longitudine și latitudine în care se consideră că locul în care se află utilizatorul nu s-a modificat). De asemenea, pentru fiecare intrare din baza de date este stocat numele locației, data și ora sosirii și ora la care utilizatorul a părăsit acea locație.

Interogarea bazei de date se face în limbaj natural. Pentru aceasta, întrebarea este transformată în text folosind modulul de speech-to-text dezvoltat în cadrul proiectului component TADARAV. Ulterior, întrebarea este parsată pentru a fi extrase informații temporale. În acest scop, s-a folosit identificarea și filtrare părților de vorbire (de ex. adverbele temporale s-au dovedit foarte utile), precum și un set de șabloane realizate manual. Astfel sunt identificate cu ușurință expresii de genul "ieri dimineață", "acum trei ore", "mai devreme" etc.

Următorul pas este identificarea referințelor la locații din întrebare, folosindu-se un gazetter, alături de șabloane de identificare a numelor de locații. Numele locațiile pot diferi parțial de cele stocate în baza de date, din acest motiv algoritmul de potrivire se bazează pe potrivirea parțială și pe locațiile frecventate mai des. Astfel, dacă utilizatorul întreabă "La ce oră am fost azi la poștă?", dintre toate oficiile poștale este selectat cel în care a fost utilizatorul în ultima zi, iar în caz de ambiguitate (mai multe oficii vizitate), este luat cel la care a mai fost utilizatorul anterior. De asemenea sunt considerate sinonime ale locațiilor identificate în întrebare, de ex. dacă utilizatorul întreabă despre o „prăvălie", se ajunge prin setul de sinonime din WordNet la "magazin". O altă ambiguitate întâlnită a fost cea pentru prepoziția "la", care poate introduce atât o entitate timp ("la ora 14:30", "la amiază"), cât și o entitate de loc ("la magazin").

După identificarea răspunsului în baza de date, aplicația folosește alt set de șabloane pentru a formula răspunsul. Șabloanele se bazează mult de cuvintele cheie din întrebare. Au fost identificate 8 tipuri de răspunsuri așteptate de utilizator, 3 pentru locații și 5 pentru informații temporale. Sunt considerate de asemenea întrebări care se referă la intervale, nu doar data/locație fixă.

Ultimul pas este transformarea textului în voce folosind API-ul aplicației dezvoltate în cadrul proiectului SINTERO, înainte de a-i fi oferit utilizatorului răspunsul.

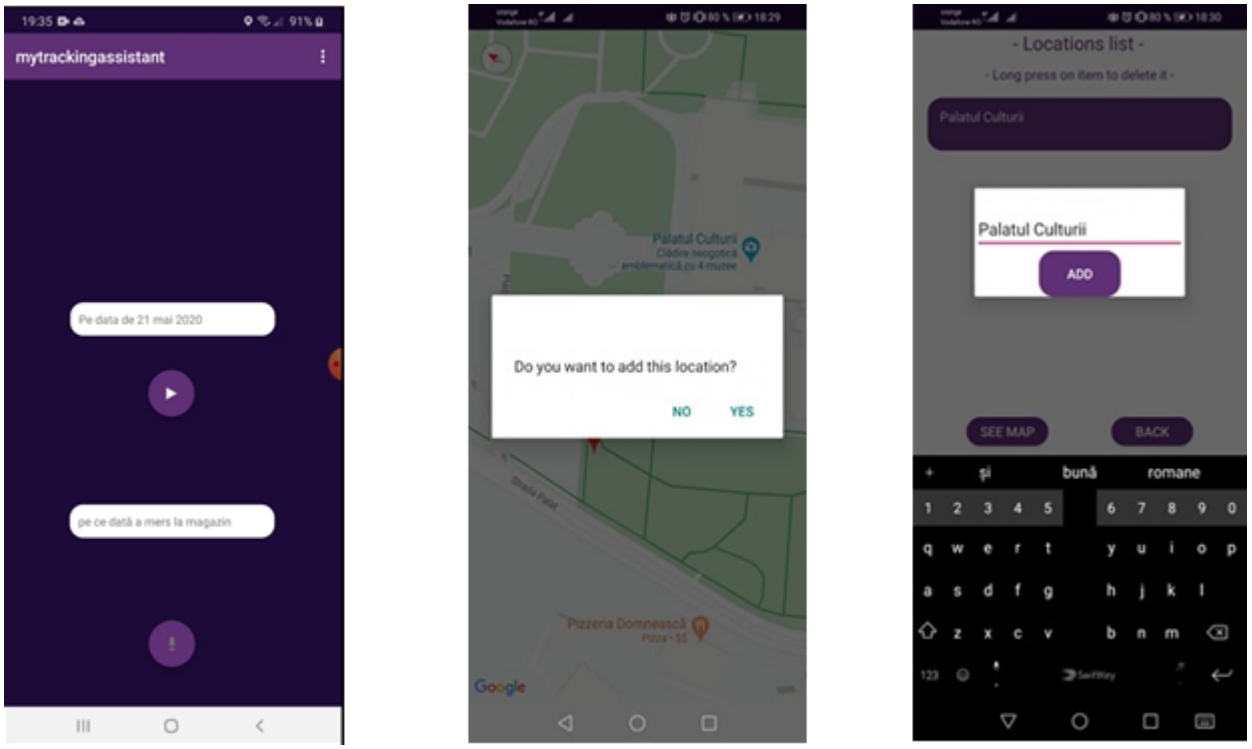

Fig. 5. Exemple de folosire a aplicației Tracking Assistant

Câteva exemple de întrebări la care aplicația răspunde sunt:

- Unde (am fost, m-am aflat, eram, mă aflam, etc.) la ora 18:00?

Aplicația returnează răspunsul pentru cel mai apropiat moment din timp care are o intrare. De exemplu, dacă utilizatorul pune această întrebare la ora 19.00, va fi căutat răspunsul între ora 17.58-18.02. Dacă în schimb utilizatorul pune întrebarea la ora 15.00, aplicația va căuta locația în intrările pentru ziua precedentă

- Unde (am fost, m-am aflat, eram, mă aflam, etc.) azi/ieri/alaltăieri/luni/marți/pe 19.10 (la 14:00)?

Dacă întrebarea include adverbele sau substantive temporale, ele sunt transformate în data specifică. Dacă nu este specificată ora, este mereu returnată ultima intrare găsită pentru ziua respectivă.

Sunt acceptate și alte formule introductive de genul Pe unde, În ce loc, În ce locație, Care era locația mea, Cum se numea locul unde, etc.

- Când am fost la magazin? Pe ce dată am fost la magazin? Pe la ce oră/Pe la cât am fost (ieri/marți/alaltăieri) la magazin?

Similar cu întrebările pentru locație, dacă nu este specificată data este returnată ultima intrare găsită pentru “magazin”.

- Între ce ore am fost (ieri) la facultate? Între cât și cât am stat acasă?

Pentru întrebările care presupun un interval, sunt căutate toate intrările din baza de date pentru intervalul respectiv, și sunt returnate grupate.

Dezvoltările ulterioare avute în vedere se axează pe adăugarea unei funcții pentru parsarea corectă a întrebărilor în care o entitate de loc/timp este dependentă de o altă entitate (de ex. “Unde am fost înainte să merg la magazin?”), precum și o funcționalitate de vizualizare a traseului zilei.

3.8 Concluzii

Aplicațiile descrise mai sus arată potențialul pe care colecțiile de resurse bimodale de genul celei disponibile pe platforma CoBiLiRo și a tehnologiilor realizate în proiecte componente le pot avea în realizarea de aplicații cu impact economic, didactic, turistic ș.a.m.d.

O parte din ideile prezentate sunt în diverse stadii de dezvoltare, ca proiecte de laborator, unele (cum ar fi aplicațiile 3, 4 și 6) având prototipuri deja funcționale.

Toate obiectivele incluse în plan la această activitate au fost realizate.

Bibliografie

[1] I. Radu, Raport Activitate A1.5: *Definirea specificațiilor funcționale și arhitecturale ale platformei integrate și configurabile de prelucrare a textelor*, proiectul ReTeRom

[2] I. Radu, Raport Activitate A1.6: *Definirea modulelor software și a serviciilor oferite de proiect; identificarea adaptărilor pentru modulele NLP existente și a modulelor noi necesare,* proiectul ReTeRom.

[3] T. Boroș, Ș. Dumitrescu, V. Pais: *Tools and resources for Romanian text-to-speech and speech-to-text applications*, 2018.

[4] A.N. Zamfirescu, T.E. Rebedea, *Identificarea entităţilor, citatelor şi evenimentelor în ştiri şi texte din Web-ul social în limba română*, în Revista Română de Interacţiune Om-Calculator 6 (2) 2013, 169-192.

[5] C. Burileanu, H. Cucu, Raport Activitate A1.11: *Studiul metodelor din literatură pentru alinierea transcrierilor aproximative cu semnalul de vorbire*, proiectul ReTeRom

[6] Aldabbagh, Omar & Mohsen, Khalid. (2014): Design and Implementation an Online Location Based Services Using Google Maps for Android Mobile. International Journal of Computer Networks and Communications Security. 2. 113-118;