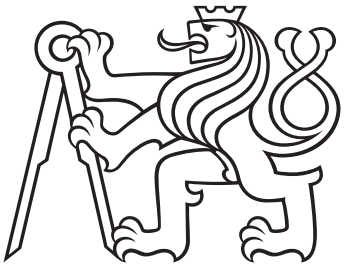
České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídící techniky
Studijní program: Kybernetika a robotika



Rozpoznávání řeči s dostupnými internetovými moduly

Speech Recognition Based on Available Internet Modules

BAKALÁŘSKÁ PRÁCE

Vypracoval: Adam Jirkovský
Vedoucí práce: Doc. Ing. Petr Pollák, CSc.
Rok: 2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Jirkovský** Jméno: **Adam** Osobní číslo: **483628**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Rozpoznávání řeči s dostupnými internetovými moduly

Název bakalářské práce anglicky:

Speech Recognition Based on Available Internet Modules

Pokyny pro vypracování:

1. Seznamte se s principy rozpoznávání spojité řeči a vypracujte stručný přehled používaných přístupů (GMM-HMM, DNN-HMM, End-to-End).
2. Proveďte rešerši veřejně dostupných modulů pro realizaci rozpoznávání spojité řeči pro češtinu a zvažte možnosti jejich použití.
3. Na platformě standardního PC rozpoznávač spojité řeči implementujte a analyzujte dosaženou úspěšnost rozpoznávání na vytvořené referenční testovací množině i při variantě on-line provozu vytvořeného rozpoznávače. Zvažte možnost použití pro přepis off-line audio či audio-video záznamů.

Seznam doporučené literatury:

[1] X. Huang, A. Acero, H.-W. Hon. Spoken Language Processing. Prentice Hall, 2001.
[2] D. Yu, L. Deng. Automatic Speech Recognition A Deep Learning Approach.Springer-Verlag London. 2015.
[3] Google. Cloud Speech-to-Text . [on-line] https://cloud.google.com/speech-to-text
[4] J. Psutka, L. Müller, J. Matoušek, V. Radová. Mluvíme s počítačem česky. Academia 2006.
[5] J. Uhlíř, a kol.: Technologie hlasových komunikací. Nakladatelství ČVUT, Praha, 2007.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Petr Pollák, CSc., katedra teorie obvodů FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **15.01.2021** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce:
do konce letního semestru 2021/2022

doc. Ing. Petr Pollák, CSc.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ _____

Datum převzetí zadání Podpis studenta

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne .....

.....
Adam Jirkovský

Poděkování

Děkuji vedoucímu práce Doc. Ing. Petru Pollákovi, CSc. za cenné rady a ochotnou pomoc při tvorbě této bakalářské práce.

Adam Jirkovský

Název práce:
Rozpoznávání řeči s dostupnými internetovými moduly

Autor: Adam Jirkovský

Studijní program: Kybernetika a robotika
Druh práce: Bakalářská práce

Vedoucí práce: Doc. Ing. Petr Pollák, CSc.
Katedra teorie obvodů

Abstrakt: Tato práce prozkoumává možnosti využití on-line modulů pro převod řeči do textu při tvorbě aplikace sloužící k přepisu obsahu multimediálních souborů. Jsou zde stručně popsány základní principy využívané v automatickém rozpoznávání řeči a shrnuty vlastnosti jednotlivých dostupných rozpoznávačů, zahrnující jejich funkce, podporované jazyky a finanční podmínky použití. Pro vývoj aplikace byl zvolen modul Google Cloud Speech-to-Text API. Způsob jeho použití a dostupné funkce jsou popsány podrobněji a je vyhodnocena dosažená kvalita přepisu promluv v českém jazyce. Závěrečná část práce je věnována popisu vývoje realizované aplikace, jejímu použití a implementovaným funkcím.
Klíčová slova: automatické rozpoznávání řeči, ASR, hluboké neuronové sítě, DNN, internetové moduly, přepis multimediálního záznamu, Python, Google Cloud Speech-to-Text API

Title:
Speech Recognition Based on Available Internet Modules

Author: Adam Jirkovský

Abstract: This thesis explores the possibilities of using online speech-to-text engines to develop an app used for multimedia content transcription. A brief description of fundamental principles used in automatic speech recognition is provided, as well as a summary of available online recognizers, their features, supported languages and usage fees. Google Cloud Speech-to-Text API was chosen for the development task. Its usage and available features are described in more detail and its performance on czech-spoken utterances is evaluated. The final part of the thesis describes development of the app, its features and usage.
Key words: automatic speech recognition, ASR, deep neural networks, DNN, internet engines, multimedia transcription, Python, Google Cloud Speech-to-Text API

Obsah

Seznam použitých zkratek

API Application Programming Interface
ASR Automatic Speech Recognition
CMU Carnegie Mellon University
CTC Connectionist Temporal Classification
DNN Deep Neural Network
GMM Gaussian Mixture Model
HMM Hidden Markov Model
HTK Hidden Markov Model Toolkit
HTML Hypertext Markup Language
LVCSR Large Vocabulary Continuous Speech Recognition
PCM Pulse-code Modulation
WER Word Error Rate

Seznam tabulek

Seznam obrázků

Kapitola 1

Úvod

Automatické rozpoznávání řeči (ASR), často také označované jako převod řeči do textu, je kybernetická disciplína zabývající se automatickou transformací audiozáznamu mluveného slova do textové podoby. Přestože první přístroje schopné rozpoznávat jednotlivá slova začaly vznikat již v 50. letech 20. století [1, 2], největší rozmach této technologie zažíváme právě nyní. Vděčíme za to nejen stále přibývajícím pokrokům v oblastech strojového učení a neuronových sítí, ale také rostoucí poptávce po službách, které na této technologii přímo závisí. Mezi ně patří například stále se zdokonalující hlasové asistenty, které jsou již svým řečovým projevem a rychlostí reakce schopné budit dojem komunikace se skutečným člověkem [3]. Funkce hlasového ovládání je standardem také v infotainment systémech moderních automobilů, kde umožňuje bezpečně ovládat satelitní navigaci nebo obsluhovat mobilní telefon. Mimo to může rovněž sloužit jako cenná kompenzační pomůcka pro hendikepované uživatele, ať už trpí problémy se zrakem, nebo nějakou formou pohybového znevýhodnění.

První přístroj schopný samostatně rozpoznávat vyřčená slova vznikl v roce 1952 v Bellových laboratořích v USA. Po přizpůsobení na míru konkrétního řečníka byl schopen rozlišit jednociferné číslovky s přesností přibližně 98% [4]. Na tento úspěch navázala společnost IBM sestrojením přístroje s názvem Shoe-box, který kromě jednotlivých číslovek rozeznával také názvy jednoduchých matematických operací, díky čemuž mohl být využit jako hlasově ovládaná kalkulačka [5]. V průběhu 50. a 60. let vznikaly v USA, Japonsku a Velké Británii také první rozpoznávače jednotlivých slabik a hlásek, a byly tak položeny základy pro rozpoznávání souvislé řeči [1].

V 70. letech začala výzkum v této oblasti podporovat výzkumná agentura amerického ministerstva obrany ARPA (dnes DARPA). Jedním z plodů tohoto rozhodnutí bylo vytvoření systému Harpy na pittsburghské univerzitě Carnegie Mellon (CMU). Harpy disponoval slovníkem s více než tisíci slovy a díky použití jazykového modelu v podobě konečného stavového automatu dokázal rozpoznávat souvislé věty. Na tento úspěch pak univerzita navázala v 90. letech představením rozpoznávacího systému CMU Sphinx, který je dodnes schopen konkurovat i moderním alternativám. DARPA rovněž v 90. letech iniciovala vznik standardizovaných testů pro hodnocení úspěšnosti rozpoznávání řeči. Hlavním ukazatelem kvality rozpoznávání se stal word error rate (WER), tedy podíl špatně rozpoznaných slov vůči celkovému počtu slov ve výroku. [1]

Jako řešení problému různorodosti řečového projevu jednotlivých řečníků se během 80. let prosadily skryté Markovovy modely (HMM). V polovině 90. let pak Cambridgeská universita vytvořila základní framework pro tvorbu řečových rozpo-

znávačů nazvaný Hidden Markov Model Toolkit (HTK), který je výzkumníky hojně využíván dodnes. [1, 6]

Technologie rozpoznávání řeči se nadále rozvíjí i ve 21. století. Stále větší roli v této oblasti získávají neuronové sítě a při realizaci náročných výpočtů s nimi spojených se postupně prosazují cloudová řešení, která již dokáží přepis do textu provádět v reálném čase. Cílem teoretické části této práce je ověření dostupnosti těchto on-line služeb pro běžného uživatele, zhodnocení poskytovaných funkcí a finančních podmínek, a výběr vhodné služby pro použití ve vlastní aplikaci realizující převod řeči do textu v českém jazyce. Praktická část práce pak bude věnována samotné tvorbě této aplikace s využitím programovacího jazyka Python.

Přesto, že současná pandemická situace a s ní spojená distanční výuka značně komplikuje provoz vysokých škol, jednou z mála jejích výhod je distribuce přednášek v digitální podobě a související možnost si tyto přednášky zpětně přehrávat. Aplikace vytvořená v rámci této práce vzniká v reakci na tyto nové možnosti. Její hlavní funkcí bude přepis obsahu videozáznamu přednášek do textové podoby. Výsledný přepis pak bude možné zobrazit, vyhledávat v něm klíčové fráze a vybranou pasáž pak automaticky přehrát v integrovaném mediálním přehrávači.

Následující druhá kapitola obsahuje stručný popis základních principů využívaných k realizaci ASR a výčet dostupných on-line rozpoznávačů, které lze i bez hlubších teoretických znalostí integrovat do vlastního programu. Třetí kapitola podrobněji popisuje funkce podporované modulem Google Cloud Speech-to-Text API, způsob použití tohoto modulu k vygenerování textového přepisu videozáznamu, a zhodnocení dosažené kvality rozpoznávání. Ve čtvrté kapitole je popsán vývoj realizované aplikace, její implementované funkce a možnosti využití. Závěrečná pátá kapitola pak obsahuje shrnutí obsahu této práce.

Kapitola 2

Principy a dostupné moduly ASR

V první části této kapitoly jsou popsány základní principy analýzy řečových signálů, a především základní přístupy používané při rozpoznávání spojité řeči. Druhá část pak obsahuje rešerši dostupných modulů pro ASR, které je možné použít při tvorbě vlastní aplikace. V závěru kapitoly je vybrán modul, který lze nejsnáze použít pro kvalitní přepis technicky zaměřených přednášek prezentovaných v českém jazyce.

2.1 Principy ASR

Základní předpokladem, který lidem umožňuje efektivně komunikovat pomocí řeči je schopnost tvořit hlas. Ten vzniká rozechvíváním hlasivek pomocí proudu vzduchu hnaného z plic. Kmitající hlasivky způsobí fluktuace tlaku ve vydechovaném vzduchu a tvoří tak zvukové vlny. Základní frekvence tohoto vlnění je přímo ovlivněna fyzickými vlastnostmi hlasivek a projevuje se v tónu hlasu konkrétního řečníka. Díky rezonanci v nosních dutinách a dutině ústní dochází k dalšímu zesílení vzniklého zvuku a díky artikulaci s pomocí jazyka, rtů a dalších částí ústní dutiny pak vznikají jednotlivé hlásky. Ty se projevují krátkodobými změnami ve frekvenčním složení generovaného zvuku a jsou základním nositelem jeho významu. [7, 8]

Při digitálním zpracování zvukového signálu dochází k zachycení zvukových vln na membráně mikrofonu, která svými kmity vyvolá vznik analogového elektrického signálu. Ten je následně vzorkován a kvantován pomocí pulzně kódové modulace (PCM). Pro účely rozpoznávání řeči se typicky využívá vzorkovacího kmitočtu 16 kHz, který poskytuje vhodný kompromis mezi rozlišitelností hlasových charakteristik a množstvím zdrojů potřebných pro přenos a uchování dat. Pro kvantování se obvykle používají rozsahy 16 nebo 24 bitů. [7]

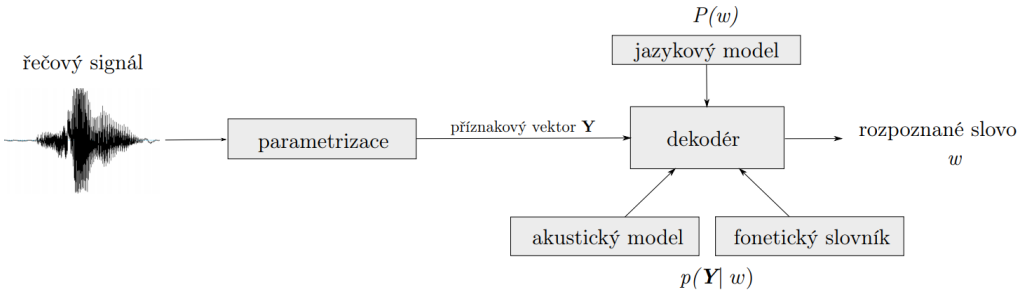
V této podobě se však záznam pro rozpoznávání řeči typicky nepoužívá. Lidský sluch totiž nevnímá časový průběh zvukového vlnění, ale pouze jeho frekvenční složení. Změny frekvence navíc nejsou vnímány lineárně a zdánlivé intenzity zvuku pro různé frekvence o stejném výkonu se liší. Na zvukový záznam je proto po částech aplikována Fourierova transformace a vzniklá spektrální reprezentace je dále transformována do jednotek mel, které reflektují nelinearitu lidského vnímání frekvence. Z výsledného Mel spektra se pomocí logaritmu a zpětné Fourierovy transformace často získávají takzvané kepstrální koeficienty, které jsou typickým vstupem pro rozpoznávače na bázi skrytých Markovových modelů. [7, 8]

Mezi dvě rozdílné disciplíny na poli rozpoznávání řeči patří rozpoznávání jednotlivých slov a rozpoznávání spojité řeči. Jednotlivá slova jsou typicky rozpoznávána při hlasovém ovládání elektronických zařízení, jako jsou například systémy chytré domácnosti nebo palubní počítače automobilů. V této situaci uživatel zpravidla precizně vyslovuje jednotlivá slova a odděluje je zřetelnými pauzami. Množství možných povelů je zároveň omezeno, což nadále usnadňuje jejich rozpoznávání. Výrazně složitější je pak úloha rozpoznávání spojité řeči a zejména její varianta s velkým slovníkem (LVCSR). Při běžné řeči totiž často dochází k provazování jednotlivých slov, chybám ve výslovnosti, či různým přeřeknutím a opravám. Velký slovník pak zvyšuje pravděpodobnost záměny jednotlivých slov. Dalším komplikujícím faktorem je odlišnost projevů jednotlivých mluvčích daná tónem hlasu, tempem řeči, intonací, přízvukem, či různými řečovými defekty. Následující sekce jsou věnovány základním moderním přístupům, které jsou schopné výzvy spojitého rozpoznávání řeči překonávat. [8]

2.1.1 Systémy na bázi HMM

Skryté Markovovy modely se jako řešení problému různorodosti řečového projevu prosadily již v 80. letech minulého století, a při realizaci rozpoznávání spojité řeči jsou do dnes často používány. Jejich ideou je využití Markovových řetězců s nepozorovatelnými stavy pro modelování posloupností řečových elementů. V případě jednodušších rozpoznávačů s omezeným slovníkem mohou být takto modelována celá slova, případně i jejich skupiny. Při LVCSR by však tento přístup vyžadoval enormní množství prostředků, a proto jsou vytvářeny modely jednodušších řečových prvků. Typicky se jedná o tzv. trifóny, které reprezentují jednotlivé hlásky v kontextu hlásky předchozí a následující. [8]

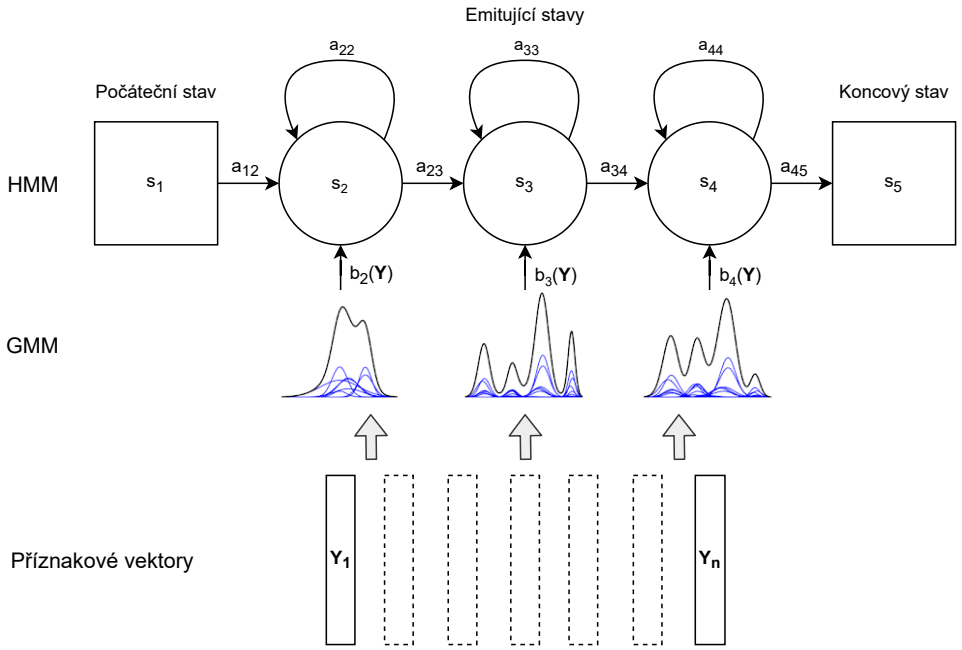
Podle principiálního schématu zachyceného na obrázku 2.1 využívá dekodér rozpoznávače na bázi HMM tři samostatné dílčí bloky. Jazykový model slouží k reprezentaci pravděpodobností výskytu jednotlivých výrazů. Při LVCSR jsou tyto pravděpodobnosti typicky modelovány pro dvojice slov, tzv. bigramy. Fonetický slovník obsahuje přepis jednotlivých slov do skupin hlásek. Pokud pro jedno slovo existuje více variant fonetického přepisu, obsahuje slovník také jejich pravděpodobnosti. Akustický model pak slouží ke klasifikaci jednotlivých trifónů na základě příznakových vektorů získaných parametrizací řečového signálu. Podle způsobu realizace akustického modelu lze systémy na bázi HMM dále dělit na varianty GMM-HMM a DNN-HMM. [8]



Obrázek 2.1: Schéma rozpoznávače na bázi HMM [8]

GMM-HMM

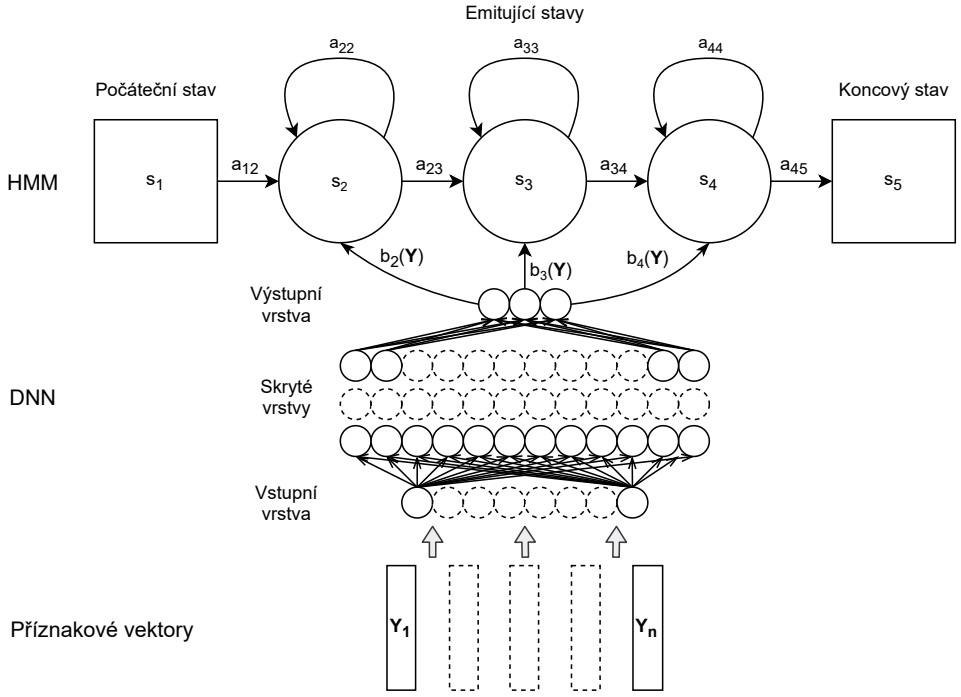
V případě GMM-HMM systémů jsou pro modelování pravděpodobnosti výskytu konkrétního příznakového vektoru pro daný stav používány tzv. Gaussovské směsi (GMM). Ty jsou vytvořeny váženým součtem několika mnoharozměrných Gaussovských pravděpodobnostních rozdělení. Díky tomu je možné zachytit různé varianty příznakových reprezentací téhož stavu. Pravděpodobnost výskytu různých variant je pak reflektována vahami jednotlivých složek. Tento princip je ilustrován na obrázku 2.2. Konkrétní trifón je zde reprezentován pětistavovým HMM, přičemž počáteční a koncový stav slouží pouze k řetězení jednotlivých modelů. Parametry a_{ij} značí pravděpodobnosti přechodu mezi jednotlivými stavy a $b_i(\boldsymbol{Y})$ značí pravděpodobnost výskytu konkrétního vektoru pozorování pro daný stav. [8]



Obrázek 2.2: Schéma akustického modelu na bázi GMM-HMM

DNN-HMM

Modernějším přístupem pro realizaci akustického modelu systému na bázi HMM je využití vícevrstevných plně propojených neuronových sítí, které bývají také označovány jako hluboké neuronové sítě (DNN). Jak lze vidět na obrázku 2.3, neuronová síť zde zastupuje modely Gaussovských směsí při výpočtu pravděpodobností pozorovaných příznaků pro jednotlivé stavy HMM. Každému z těchto stavů odpovídá jeden z výstupů této sítě. Díky tomuto přístupu lze dosáhnout lepší kvality rozpoznávání při současném snížení počtu parametrů akustického modelu. Nevýhodou je ovšem potřeba většího množství dat pro natrénování sítě. [8, 9]

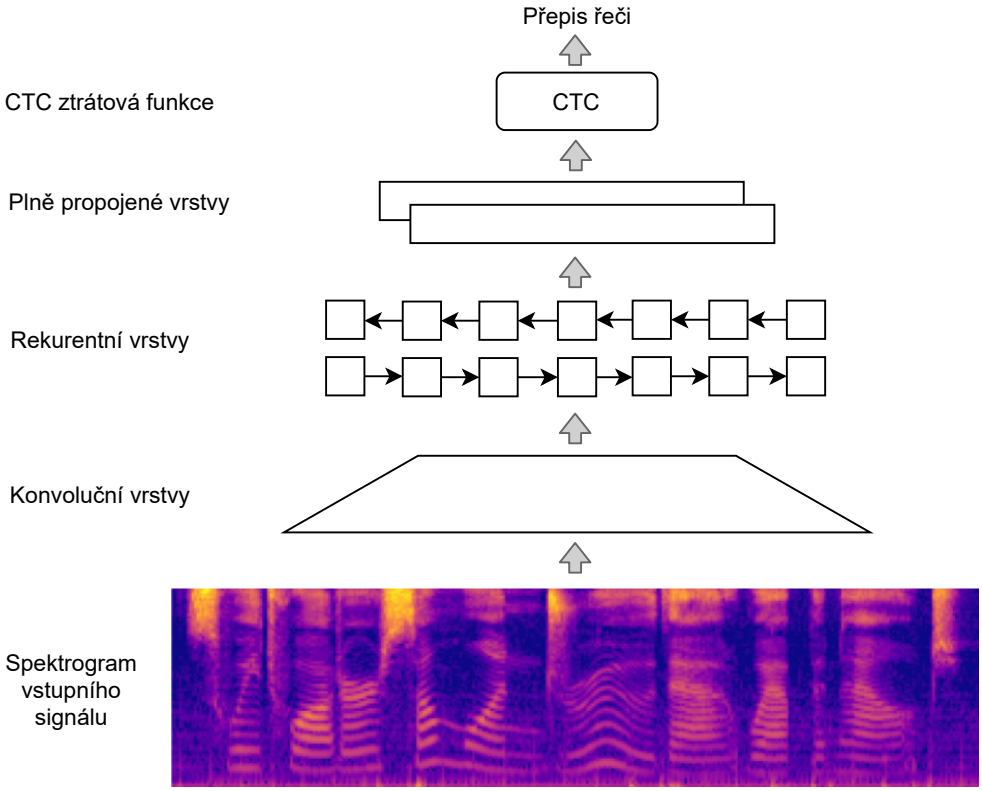


Obrázek 2.3: Schéma akustického modelu na bázi DNN-HMM

2.1.2 End-to-End systémy

V nejmodernějších systémech pro rozpoznávání řeči již není využití neuronových sítí omezeno pouze na akustický model. V tzv. End-to-End systémech jsou s jejich pomocí realizovány všechny funkční bloky rozpoznávače, parametrizaci signálu nevyjímaje. Celý systém je tak tvořen jednou neuronovou sítí, do níž vstupuje přímo digitalizovaný zvukový signál, případně jemu odpovídající spektrogram, a výstupem je textový přepis daného záznamu. Parametrizace signálu je zejména v případě zpracovávání spektrogramů realizována pomocí konvolučních vrstev. Akustický a jazykový model jsou v principu nahrazeny kombinací plně propojených a rekurentních vrstev. Eliminaci nežádoucích duplikovaných znaků na výstupu pak zajišťuje tzv. CTC ztrátová funkce. Ideové schéma End-to-End systému je zachyceno na obrázku 2.4. [8]

Díky tomu, že není třeba vytvářet modely jednotlivých řečových elementů, je tvorba End-to-End systémů v principu jednodušší než u modelů založených na HMM. Díky stále se zdokonalujícím technikám návrhu a trénování neuronových sítí lze zároveň očekávat, že se budou nadále zlepšovat i výsledky takto implementovaných systémů. Větší důraz na použití neuronových sítí s sebou však opět přináší nutnost zajištění velkého množství dat pro jejich natrénování. V aplikacích, kde je objem trénovacích dat omezen jsou proto stále s výhodou využívány systémy na bázi HMM. [8]



Obrázek 2.4: Schéma End-to-End systému

2.2 Dostupné moduly pro rozpoznávání spojité řeči

Pro usnadnění implementace výše popsaných principů při návrhu řečových rozpoznávačů je k dispozici řada nástrojů, které jsou obvykle distribuovány jako open-source software. Pro návrh a trénování systému na bázi HMM lze využít například Hidden Markov Model Toolkit vyvíjený Cambridgeskou univerzitou nebo systém Sphinx vyvíjený univerzitou Carnegie Mellon v Americkém Pittsburghu [6, 10]. Novější alternativou k těmto tradičním nástrojům je rychle se rozvíjející systém Kaldi [11]. Pro tvorbu End-to-End systémů lze pak využít nástroj Deep Speech vyvíjený společností Mozilla [12, 13]. Tyto nástroje jsou vhodné k implementaci off-line rozpoznávačů spojité řeči. Pro většinu méně rozšířených jazyků včetně češtiny je však nutné natrénování vlastních modelů, což kromě dodatečných znalostí vyžaduje také přístup k dostatečnému množství dat.

Alternativou k těmto off-line nástrojům jsou webové služby pro rozpoznávání řeči, jejichž nabídka se rychle rozrůstá. Jejich výhodou je snadné použití, které nevyžaduje odborné znalosti problematiky ASR. Pro obecné promluvy neobsahující neobvyklé výrazy u nich lze zároveň očekávat lepší kvalitu rozpoznávání než u off-line systémů trénovaných na omezeném množství dat. Nevýhodou je naopak nutnost připojení k internetu, menší míra kontroly nad ochranou uživatelských dat, a také časté zpoplatnění těchto služeb. Následující podsekce budou věnovány některým zástupcům těchto rozpoznávačů, u nichž budou kromě dostupných funkcí zmiňovány také finanční podmínky jejich použití a podpora různých jazyků.

2.2.1 Watson Speech to Text

Společnost IBM je jedním z tradičních průkopníku na poli automatického rozpoznávání řeči. Není proto divu, že jako součást cloudové platformy IBM Watson nabízí také službu pro převod řeči do textu. Mezi podporované funkce se řadí automatická detekce klíčových slov, rozlišování jednotlivých řečníků v dialogu a časové kótování jednotlivých slov. V současné době je podporováno méně než 20 světových jazyků a čeština mezi nimi bohužel nefiguruje [14]. Dodatečné funkce pro filtrování nevhodných výrazů a osobních dat jsou navíc dostupné pouze pro anglický a japonský jazyk. Výhodou může být naopak možnost přetrénovat základní modely rozpoznávače s využitím vlastních dat [15].

V případě přepisování méně než 500 minut záznamu měsíčně s využitím základních modelů lze službu využívat bezplatně. Standardní cena za minutu zpracovaného záznamu je $0,01 nebo $0,02 v závislosti na celkovém počtu minut zpracovaných za měsíc. K dispozici jsou také prémiové varianty předplatného zahrnující zvýšenou ochranu dat a možnost využívat službu pro přepis řeči i v systémech realizovaných na konkurenčních cloudových platformách. Ceny těchto předplatných jsou sjednávány individuálně.

2.2.2 Microsoft Azure Speech to Text

Microsoft je další velkou technologickou společností poskytující vlastní službu pro přepis řeči do textu. Ta je součástí cloudové platformy Microsoft Azure. Mezi podporované funkce patří například filtrování nevhodných výrazů, automatické doplňování interpunkce, rozlišování jednotlivých řečníku, analýza kvality výslovnosti ve srovnání s rodilým mluvčím a časové kótování na úrovni jednotlivých slov. K dispozici je také diktovací režim, který umožňuje vkládat interpunkční znaky pomocí klíčových slov. Služba podporuje několik desítek jazyků včetně češtiny [16]. U nejpoužívanějších světových jazyků lze navíc přetrénovat model rozpoznávače poskytnutím vlastních anotovaných dat a zlepšit tak kvalitu rozpoznávání [17].

Cena za přepis jedné hodiny zvukového záznamu se pohybuje mezi $1 a $2,40, v závislosti na aktivovaných funkcích, přičemž prvních pět hodin záznamu lze každý měsíc přepsat zdarma. Noví uživatelé mohou využít třicetidenního evaluačního období s volným kreditem $200.

2.2.3 Google Cloud Speech-to-Text API

Služba Speech-to-Text poskytovaná platformou Google Cloud využívá pokročilé architektury neuronových sítí trénované na velkém množství dat, ke kterým má společnost Google díky ostatním nabízeným službám zajištěn přístup. Jedná se proto o jeden z nejmodernějších systémů pro rozpoznávání řeči, který je stále doplňován o nové funkce, mezi něž patří např. integrovaná redukce šumu, nebo beta verze rozlišování jednotlivých řečníků v dialogu a automatického doplňování interpunkce. Dále lze aktivovat časové kótování jednotlivých slov v záznamu a zvýšit kvalitu rozpoznávání poskytnutím seznamu méně obvyklých slov, které se v záznamu mohou vyskytnout [18]. V současné době je podporováno více než 120 jazyků a dialektů, češtinu nevyjímaje [19].

První hodina přeloženého záznamu je každý měsíc zdarma, dále je za každou minutu účtováno $0,016 až $0,036, podle zvolených funkcí. Cena také závisí na udě-

lení souhlasu se sběrem dat z přepisovaného audia. Pro nové uživatele je k dispozici devadesátidenní evaluační období s volným kreditem $300. Vzdělávací instituce mohou také zažádat o kredit v hodnotě $50 pro pedagogy a studenty registrovaných kurzů.

2.2.4 Amazon Transcribe

Společnost Amazon také nezůstává pozadu s nabídkou vlastního cloudového řešení pro převod řeči do textu, které je součástí platformy Amazon Web Services. Podporované funkce zahrnují automatické doplňování interpunkce, časové kótování jednotlivých slov, rozeznávání jednotlivých řečníků, filtrování nevhodných výrazů a osobních dat a automatické rozpoznávání jazyka [20]. Rovněž je možné využít automatické extrakce zvukové stopy z videozáznamů ve fomátech WebM a MP4. Jazykový model rozpoznávače je možné natrénovat na vlastním textovém korpusu a zlepšit tak kvalitu rozpoznávání. Pro užití v medicínském prostředí je navíc k dispozici speciální varianta Amazon Transcribe Medical. Seznam podporovaných jazyků obsahuje několik desítek položek, mezi nimiž však nefiguruje čeština [21].

Cena za minutu zpracovaného záznamu se pohybuje mezi $0,008 a $0,024 v závislosti na měsíčním objemu zpracovaných dat. Použití některých pokročilých funkcí může být účtováno samostatně. Během prvních 12 měsíců využívání služby je prvních 60 minut záznamu každý měsíc zpracováno bezplatně.

2.2.5 Wit.ai

Platforma Wit.ai, vlastněná společností Facebook, poskytuje uživatelům možnost vytvářet aplikace s funkcemi virtuálního asistenta a různé chatovací automaty. Velká část jejích funkcí je proto zaměřena na zpracování již rozpoznaného textu. Součástí je ovšem i integrovaný rozpoznávač řeči, který lze samostatně využít. Jeho funkce jsou ovšem omezeny na pouhý přepis záznamu, který svou délkou navíc nesmí přesáhnout hranici 20 sekund [22]. Podporovaných jazyků je více než 30, ovšem čeština mezi ně bohužel nepatří [23]. Jedná se tak o rozpoznávač s velmi omezenou oblastí využití, jehož hlavní výhodou je bezplatný provoz. Jediným nutným závazkem je vyžadovaný účet na sociální síti Facebook.

2.3 Výběr vhodného modulu a vývojové platformy pro tvorbu aplikace

Souhrnné srovnání jednotlivých cloudových rozpoznávačů je zachyceno v tabulce 2.1. Stěžejním požadavkem pro přepis většiny přednášek na naší fakultě je podpora českého jazyka. Ta je dostupná pouze u platforem Google Cloud a Microsoft Azure. Obě tyto služby podporují časové kótování jednotlivých slov, které je pro funkci aplikace zásadní, a automatické doplňování interpunkce, které lze s výhodou použít pro usnadnění orientace v textu. Trénování vlastního modelu rozpoznávače není předmětem této práce, a proto nebyly související funkce při výběru zohledněny. Naopak možnost adaptovat slovník rozpoznávače doplněním očekávaných odborných výrazů je pro přepis technicky zaměřených přednášek velmi žádoucí. V tomto ohledu jsou oba zmíněné rozpoznávače rovněž rovnocenné. Důvodem k upřednostnění služby

Poskytovatel	IBM	Microsoft	Google	Amazon	Wit.ai
Podpora češtiny	Ne	Ano	Ano	Ne	Ne
Trénování akustických modelů	Ano	Ano	Ne	Ne	Ne
Adaptace slovníku	Ano	Ano	Ano	Ano	Ne
Detekce účastníků dialogu	Ano	Ano	Ano	Ano	Ne
Automatická interpunkce	Ne	Ano	Ano	Ano	Ne
Časové kótování slov	Ano	Ano	Ano	Ano	Ne
Min. cena za minutu záznamu [$]	0.01	0.017	0.016	0.008	0
Max. cena za minutu záznamu [$]	0.02	0.040	0.036	0.024	0

Tabulka 2.1: Srovnání cloudových služeb pro rozpoznávání řeči

Google Cloud Speech-to-Text API proto byla především dostupnost devadesátidenního evaluačního období, díky kterému bylo možné službu po celou dobu vývoje aplikace využívat zcela zdarma.

Jako vhodná vývojová platforma byl zvolen jazyk Python, který díky velkému množství specializovaných knihoven a vysoké úrovni abstrakce umožňuje vývojářům zaměřit se na klíčové funkce programu. Jedním z důvodů, proč je Python vhodný pro realizaci ASR je také dostupnost knihovny SpeechRecognition [24], kterou mohou začátečníci využít jako vstupní bránu do světa rozpoznávání řeči. Její součástí je univerzální třída reprezentující rozpoznávače od různých poskytovatelů, které lze díky tomu obsluhovat jednotným způsobem. K dispozici jsou také integrované metody pro zpracování obsahu zvukových souborů nebo nahrávek pořízených mikrofonem s využitím knihovny PyAudio [25]. Tyto metody zahrnují i možnost redukce šumu v záznamu.

Funkci rozpoznávání řeči si lze s touto knihovnou nejsnáze vyzkoušet s využitím modulu Google Speech Recognition, který nevyžaduje doinstalování dalších knihoven ani získání autentizačních údajů a plně podporuje český jazyk. Použití tohoto modulu je omezeno na 50 žádostí denně, a proto není vhodný pro využití v praxi. Díky jeho snadnému použití a bezplatné dostupnosti je však ideální variantou pro prvotní programátorské experimenty s rozpoznáváním řeči. Ukázkové skripty pro přepis lokálního zvukového souboru a záznamu pořízeného mikrofonem jsou součástí přílohy A této práce.

Knihovnu SpeechRecognition lze také využít k obsluze většiny zmíněných cloudových rozpoznávačů, které však vyžadují doinstalování dodatečných knihoven. Využití jejich pokročilých funkcí navíc již tato knihovna neumožňuje, a je proto vhodnější pracovat přímo s dedikovanými rozhraními jednotlivých rozpoznávačů.

Kapitola 3

Použití modulu Google Cloud Speech-to-Text API

Tato kapitola shrnuje poznatky získané během tvorby aplikace využívající služeb poskytovaných platformou Google Cloud. Jsou zde popsány jednotlivé funkce podporované modulem Speech-to-Text API, kroky potřebné k jejich aktivaci, a způsob jejich použití v prostředí programovacího jazyka Python s využitím potřebných knihoven.

3.1 Podporované funkce a nastavení

V závislosti na potřebách konkrétní aplikace lze při používání modulu Speech-to-Text API využít jeden ze tří základních režimů pro zpracování vstupních dat [26].

- Synchronní režim umožňuje přepis záznamu s maximální délkou jedné minuty. Tento záznam může být uložen přímo v počítači uživatele, nebo na on-line úložišti Google Cloud Storage. Po zpracování celého záznamu obdrží uživatel finální výsledek.

- Asynchronní režim umožňuje přepis až osmihodinového záznamu, který však musí být nejprve nahrán do cloudového úložiště. Během zpracovávání záznamu lze zjišťovat procentuální stav průběhu operace.

- Streamovací režim pak umožňuje přepis audia nahrávaného v reálném čase a průběžné získávání dílčích výsledků.

Zároveň je možné volit mezi čtyřmi různými modely rozpoznávače, které jsou cíleně trénovány pro zpracování konkrétního druhu záznamu.

- Model pro přepis videa je vhodný zejména pro zpracování záznamů s více střídajícími se řečníky.

- Model pro přepis telefonních hovorů je vhodný pro záznamy s nižší vzorkovací frekvencí.

- Model pro rozpoznávání hlasových povelů najde využití při přepisu kratších promluv, jako jsou hlasová vyhledávání a pokyny pro ovládání.

- Základní model je pak vhodný pro ostatní případy, například pro přepis delších záznamů promluv jednoho řečníka.

Modely pro přepis videa a telefonních hovorů jsou zpoplatněny vyššími sazbami než standardní model a pro český jazyk zatím nejsou dostupné [19].

Výsledky rozpoznávání lze nadále zdokonalovat pomocí adaptace jazykového kontextu. Požadavek na rozpoznání lze doplnit seznamem až 5000 frází, které budou preferovány před podobně znějícími alternativami. Délka jednotlivých frází je omezena na 100 znaků a celkový počet znaků nesmí přesáhnout 100 000. Frázím lze jednotlivě či po skupinách přiřadit požadovanou hodnotu relativní preference v rozmezí 0 až 20.

Nezávisle na zvoleném modelu a režimu rozpoznávání lze využité také následujících volitelných funkcí:

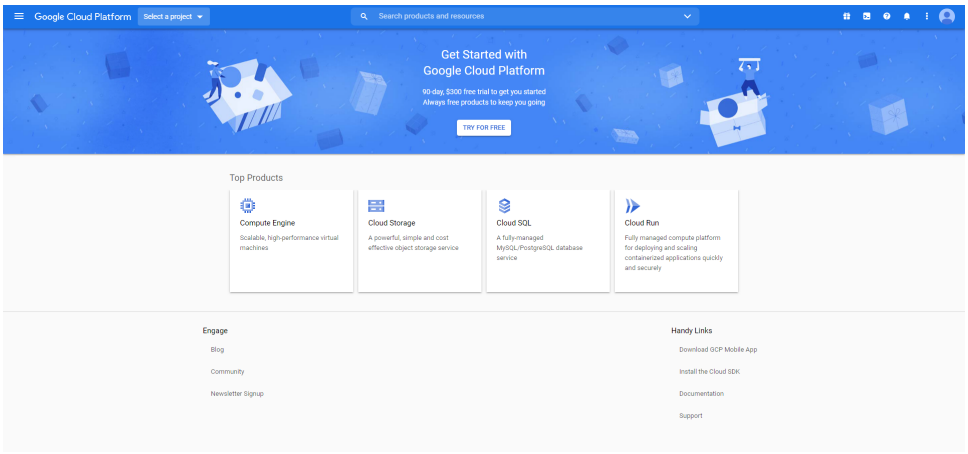
- Filtrování expresivních výrazů - Detekovaná slova jsou až na počáteční písmeno nahrazena znakem *
- Automatická detekce jazyka - Rozpoznávač vybere správný jazyk z poskytnutého seznamu až čtyř možností
- Automatické doplňování interpunkce
- Automatické rozpoznání jednotlivých řečníků
- Vytvoření časových značek pro začátky a konce jednotlivých slov
- Poskytnutí více možných výsledků rozpoznávání seřazených podle jejich věrohodnosti

Aplikace realizující přepis obsahu přednášek měla pracovat s jejich kompletními záznamy, jejichž délka výrazně přesahuje limit jedné minuty. Pro její realizaci byl proto zvolen asynchronní režim rozpoznávání. Protože během on-line přednášek nedochází s výjimkou občasných dotazů publika ke střídání řečníků a model pro přepis videa nebyl pro český jazyk dostupný, byl při tvorbě aplikace použit základní model rozpoznávače. Pro zlepšení kvality rozpoznávání technických termínů využívá aplikace možnosti adaptace jazykového kontextu. Mezi další využité volitelné funkce patří automatické doplňování interpunkce a tvorba časových značek pro jednotlivá slova.

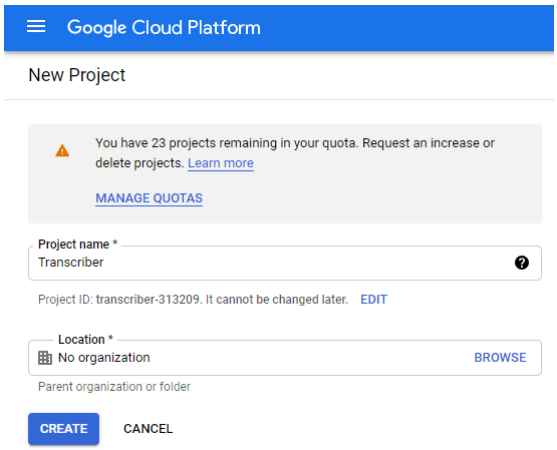
3.2 Registrace a vytvoření projektu

Služby platformy Google Cloud lze aktivovat pro existující Google účet prostřednictvím stránky [27]. Je-li k dispozici bezplatné zkušební období, lze jej aktivovat kliknutím na tlačítko „Get started for free“ v pravém horním rohu prohlížeče. Rovněž je možné kliknout na tlačítko „Console“ vlevo od ikony Google účtu a zkušební období případně aktivovat později. Po udělení souhlasu s obchodními podmínkami se zobrazí nabídka nejpoužívanějších služeb (obrázek 3.1). V levé části horní lišty lze nyní zvolit možnost „Select a project“ a v pravém horním rohu nově otevřeného dialogu kliknout na tlačítko „NEW PROJECT“. Na nové stránce (obrázek 3.2) je možné zadat název nového projektu a případně upravit jeho unikátní identifikátor. Po vytvoření projektu kliknutím na tlačítko „Create“ se v pravé horní části

stránky zobrazí upozornění o nově vytvořeném projektu. Po kliknutí na možnost „SELECT PROJECT“ uvnitř tohoto oznámení se zobrazí nástěnka se základními informacemi o projektu.

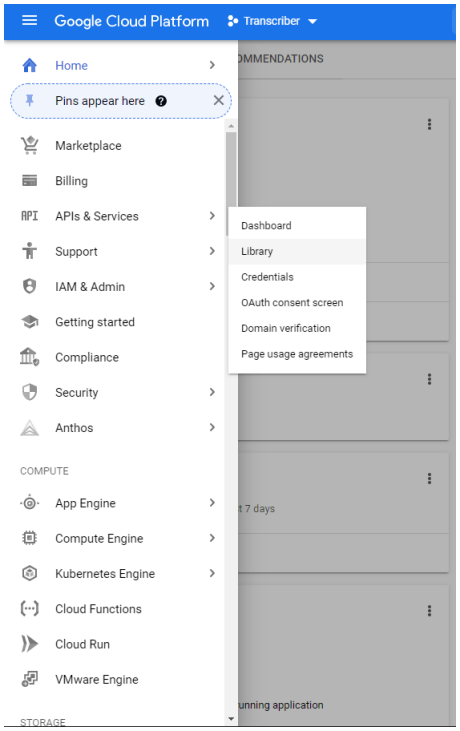


Obrázek 3.1: Úvodní stránka zobrazená v platformě Google Cloud po prvním přihlášení

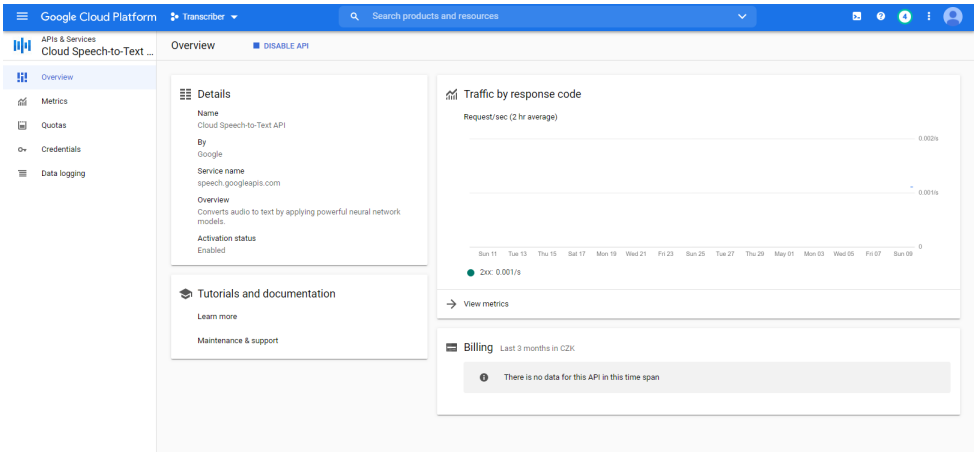


Obrázek 3.2: Založení nového projektu

Pro aktivaci služby pro přepis řeči do textu je nyní třeba otevřít sloupcové menu kliknutím na ikonu v levém horním rohu stránky. Po zvolení položky „Library“ uvnitř podmenu „APIs & Services“ (obrázek 3.3) lze do vyhledávacího pole začít psát výraz „Speech-to-Text“. Brzy se zobrazí seznam návrhů obsahující položku Cloud Speech-to-Text API. Po jejím zvolení se zobrazí informace o této službě včetně finančních podmínek jejího používání. Službu lze aktivovat kliknutím na tlačítko „ENABLE“. V případě, že dosud nebylo provedeno nastavení platebních údajů, je třeba následovat pokyny v automaticky zobrazeném dialogu. Po úspěšné aktivaci služby se zobrazí podrobné informace a statistiky jejího využívání (obrázek 3.4). V záložce „Data logging“ je dále možné povolit sběr anonymních dat za účelem trénování řečového rozpoznávače a získat tak výhodnější sazbu za používání služby.

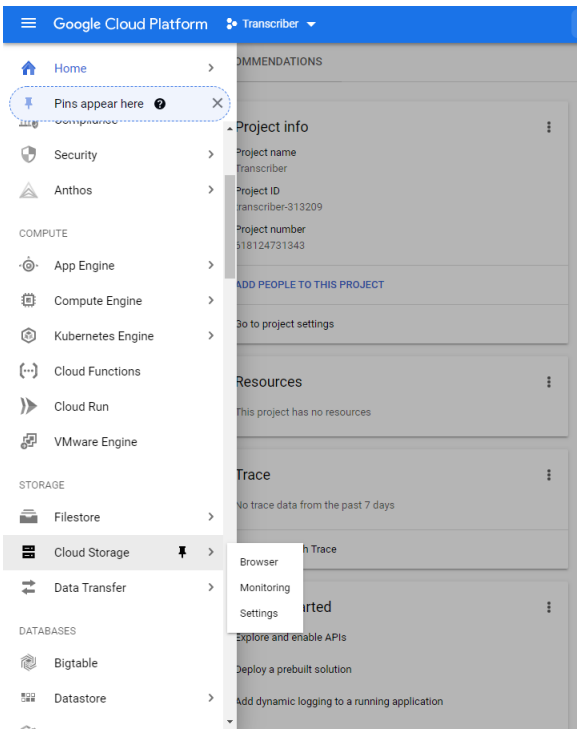


Obrázek 3.3: Umístění knihovny služeb v rozbalovacím menu



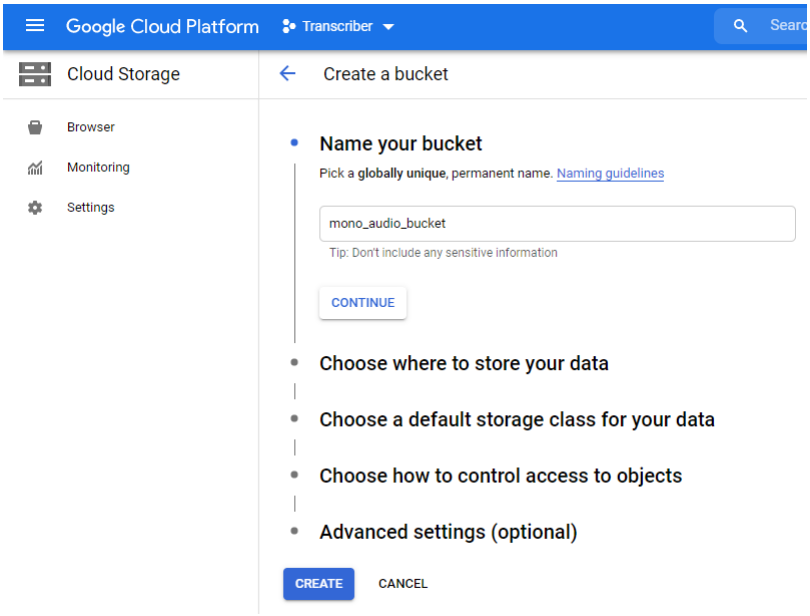
Obrázek 3.4: Stránka pro správu služby Speech-to-text

Pro přepis dlouhých zvukových záznamů je dále třeba vytvořit cloudové úložiště, kam boudu data před zahájením rozpoznávání nahrána. To lze provést výběrem položky „Cloud Storage“ v rozbalovacím sloupcovém menu (obrázek 3.5). Po kliknutí na možnost „CREATE BUCKET“ v horní části nově otevřené stránky se zobrazí formulář pro vytvoření nové úložné schránky. Po zadání požadovaného unikátního názvu (obrázek 3.6) je možné upravit i další nastavení, která ovlivní fyzické umístění úložiště v síti a jeho další parametry. Po vytvoření úložiště kliknutím na tlačítko „CREATE“ jej lze dále spravovat prostřednictvím automaticky otevřené stránky. Záložku „LIFECYCLE“ lze například využít k nastavení automatického mazání souborů po uplynutí zvoleného intervalu. Využívání cloudového úložiště je rovněž samostatně zpoplatněno. Při daných datových objemech jsou však tyto náklady ve srovnání se službou Speech-to-Text zcela zanedbatelné.

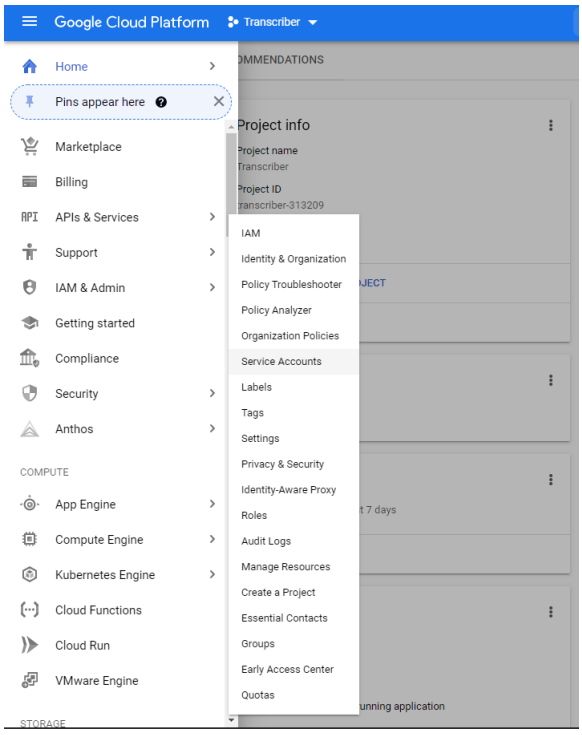


Obrázek 3.5: Umístění správy cloudového úložiště v rozbalovacím menu

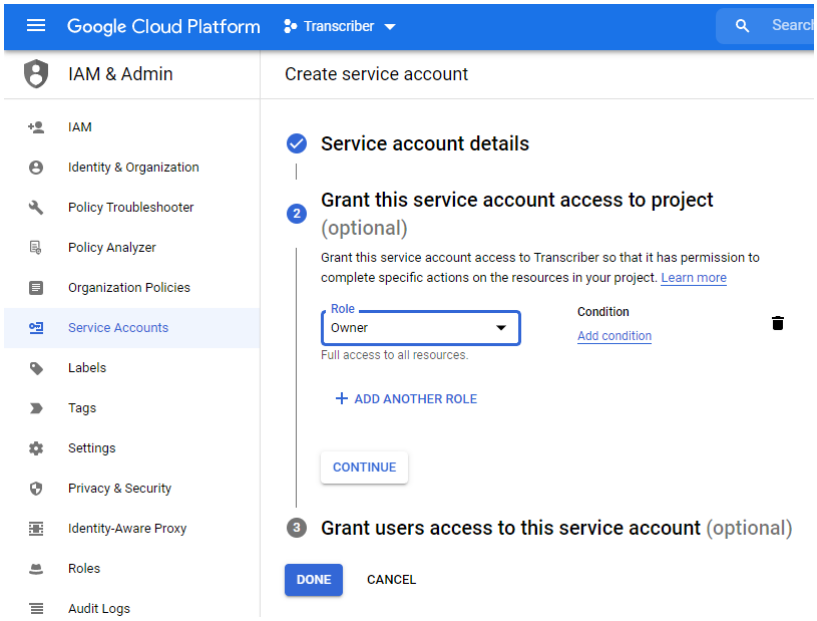
Finálním krokem je vytvoření autentizačního souboru pro přístup k aktivovaným službám. K tomu je zapotřebí v rozbalovacím menu zvolit skupinu „IAM & Admin“ a v ní položku „Service Accounts“ (obrázek 3.7). Po kliknutí na možnost „CREATE SERVICE ACCOUNT“ v horní části nově otevřené stránky se opět zobrazí vytvářecí formulář. Po vyplnění jeho první části a kliknutí na tlačítko *CREATE* je dále třeba v druhé části přidělit tomuto účtu vlastnické oprávnění ke stávajícímu projektu (obrázek 3.8). Po kliknutí na možnost „DONE“ se zobrazí seznam obsahující nově vytvořený servisní účet. Kliknutím na ikonu ve sloupci „Actions“ a výběrem položky „Manage keys“ z kontextového menu lze přejít ke správě autentizačních klíčů. Po rozbalení menu „ADD KEY“ a zvolení položky „Create new key“ (obrázek 3.9) se zobrazí dialog s volbou formátu vytvářeného klíče. Po zvolení možnosti „JSON“ a kliknutí na tlačítko „CREATE“ dojde k zahájení stahování autentizačního souboru, který lze uložit například pod názvem credentials.json. Nyní je již možné začít služby využívat prostřednictvím zvoleného programovacího prostředí.



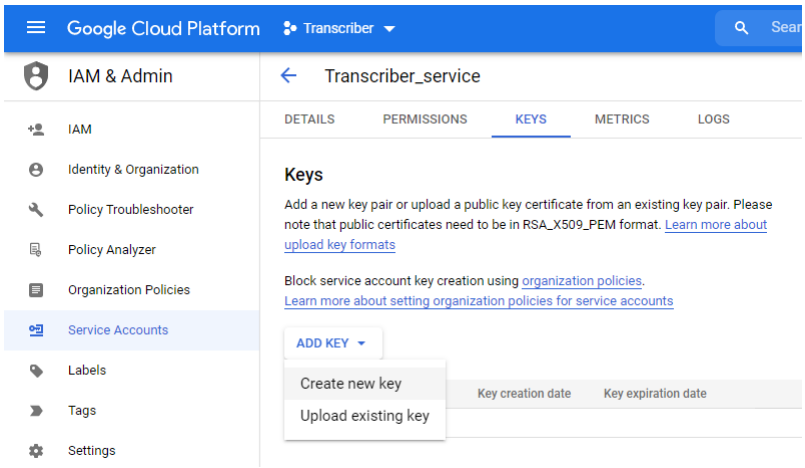
Obrázek 3.6: Vytvoření nové úložné schránky



Obrázek 3.7: Umístění správy servisních účtů v rozbalovacím menu



Obrázek 3.8: Vytvoření nového servisního účtu a udělení přístupu k existujícímu projektu



Obrázek 3.9: Správa autentizačních klíčů

3.3 Implementace v Pythonu

Pro účely navrhované aplikace jsou využívány služby Google Cloud Speech-to-Text API a Google Cloud Storage. Jejich obsluhu v jazyce Python zajišťují knihovny google-cloud-speech [28] a google-cloud-storage [29]. Obě služby vyžadují pro svou funkci tentýž autentizační soubor, jehož vytvoření je popsáno v předchozí sekci. Za předpokladu, že je tento soubor s názvem credentials.json uložen v kořenovém adresáři programu, lze jej pro tento účel zpřístupnit pomocí následujícího příkazu:

```
import os
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = 'credentials.json'
```

Kód v následující ukázce pak vytvoří objekty zajišťující funkce jednotlivých služeb. V případě, že by autentizační soubor neexistoval, nebo obsahoval chybné údaje, skončil by program s chybou.

```
from google.cloud import storage
from google.cloud import speech_v1p1beta1 as speech
storage_client = storage.Client()
speech_client = speech.SpeechClient()
```

Pro extrakci zvukové stopy z videozáznamu byla využita knihovna MoviePy [30]. Následující ukázka zachycuje postup extrakce jednokanálového audia ze souboru video.mp4 a jeho uložení do souboru audio.wav.

```
import moviepy.editor as mp
video_clip = mp.VideoFileClip('video.mp4')
audio_clip = video_clip.audio
audio_clip.write_audiofile('audio.wav', ffmpeg_params=["-ac", "1"])
```

Získanou zvukovou stopu je dále třeba nahrát na cloudové úložiště. K tomu lze využít schránku mono_audio_bucket vytvořenou v sekci 3.2. Příkazy v následující ukázce zajistí upload lokálního souboru audio.wav do této schránky, kde bude uložen pod názvem uploaded_audio.wav.

```
bucket = storage_client.get_bucket('mono_audio_bucket')
blob = bucket.blob('uploaded_audio.wav')
blob.upload_from_filename('audio.wav')
```

Nyní lze již zahájit samotné asynchronní rozpoznávání s využitím metody *long_running_recognize* třídy *SpeechClient*. V následující ukázce je prezentováno vytvoření povinných parametrů *config* a *audio* a použití této metody pro přepis souboru nahraného v předchozí ukázce.

```
audio = speech.RecognitionAudio(
        {'uri': 'gs://mono_audio_bucket/uploaded_audio.wav'})
config = speech.RecognitionConfig({
        "language_code": 'cs',
        "enable_word_time_offsets": True,
        "enable_automatic_punctuation": True,
        "speech_contexts": [
```

```
            {"phrases": ['švestka'], "boost": 0.5},
            {"phrases": ['šestka'], "boost": 2}
        ]})
operation = speech_client.long_running_recognize(
            config=config, audio=audio)
recognition_data = operation.result()
```

Konfigurační objekt obsahuje nastavení rozpoznávání v českém jazyce s aktivními volitelnými funkcemi pro automatickou interpunkci a časové kótování jednotlivých slov. Zároveň je využita funkce pro úpravu jazykového kontextu ke snížení relativní pravděpodobnosti slova „švestka“ a navýšení pravděpodobnosti pro slovo „šestka“.

Samotné volání metody *long_running_recognize* neblokuje běh programu a lze se po něm periodicky dotazovat na procentuální stav průběhu operace prostřednictvím parametru *metadata.progress_percent* objektu *operation*. Pomocí metody *done* pak lze zjistit, zda je operace již dokončena. Operaci lze také zrušit pomocí metody *cancel*, jejíž použití však v současné verzi potřebné knihovny vyvolá chybu způsobenou neimplementovanými součástmi. Tuto chybu lze ošetřit použitím bloku try-except. Vyčkání na výsledek operace lze rovněž zajistit pouhým použitím metody *result*, jak je demonstrováno v ukázce výše.

Výsledkem rozpoznávání je objekt *recognition_data* typu *LongRunningRecognizeResponse*, jehož parametr *results* obsahuje seznam jednotlivých dílčích výsledků. Každý z těchto výsledků odpovídá přibližně minutovému úseku původního záznamu, jehož rozdělení si zajišťuje rozpoznávač automaticky. Jednotlivé výsledky pak uvnitř parametru *alternatives* obsahují seznam možných variant přepisu. V případě, že nebyl uvnitř požadavku na rozpoznání uveden počet požadovaných variant, je tento seznam pouze jednopoložkový. Tyto alternativy obsahují vždy kompletní přepis odpovídajícího úseku záznamu uvnitř parametru *transcript* a pravděpodobnost tohoto přepisu v poli *confidence*. V případě požadavku na časové kótování jednotlivých slov jsou pak výsledná data uložena v poli *words*.

Kompletní přepis celého záznamu lze uložit do textového souboru transcript.txt pomocí následujících příkazů.

```
with open('transcript.txt', 'w') as transcript_file:
    for result in recognition_data.results:
        transcript_file.write(result.alternatives[0].transcript)
        transcript_file.write(' ')
```

Celý výsledek rozpoznávání lze také transformovat do textového řetězce ve formátu JSON pomocí metody *to_json* datového typu *LongRunningRecognizeResponse*. Tento řetězec lze pak uložit do textového souboru a opětovně načíst pomocí metody *from_json*.

S využitím výše popsaných funkcionalit byl vytvořen nástroj pro přepis obsahu audio a video souborů, který je uložen na kompaktním disku přiloženém k této práci (příloha B). Po spuštění skriptu transcribe_media.py s argumentem obsahujícím cestu ke zdrojovému mediálnímu souboru dojde k automatickému vytvoření dvou souborů, jejichž umístění bude vypsáno do příkazové řádky. Vytvořený textový soubor obsahuje přepis záznamu v čitelné podobě s časovým kótováním začátků jednotlivých odstavců. K jeho prohlížení je vhodné použít textový editor s automatickým zalamováním řádků. Soubor formátu JSON pak obsahuje kompletní výsledek

rozpoznávání převedený do podoby textového řetězce. Tento soubor lze využít k následnému načtení dat do aplikace popsané v následující kapitole.

3.4 Zhodnocení kvality rozpoznávání

Pro zhodnocení kvality rozpoznávání byly použity krátké nahrávky obsažené v databázích CtuTest a CzLecDSP poskytnutých vedoucím práce. První zmíněná databáze obsahuje čtené promluvy prosté odborných výrazů. Druhá databáze je pak tvořena nahrávkami z přednášek pojednávajících o zpracování zvukového signálu a rozpoznávání řeči. Obě databáze obsahují pro každou promluvu dvě nahrávky lišící se kvalitou použitého mikrofonu. Ve všech případech byly z evaluace vyřazeny nahrávky obsahující číslovky. Ty jsou rozpoznávačem Google Cloud Speech-to-Text API často zapisovány v číslicové podobě, což není kompatibilní se striktně slovním zápisem realizovaným v referenčních přepisech.

Pro databázi CzLecDSP byl zároveň k dispozici seznam slov vyskytujících se v obsažených promluvách. Z něj byla odstraněna běžná slova obsažená v databázi Českého národního korpusu CNK340 a výsledný seznam byl pak použit k adaptaci jazykového kontextu rozpoznávače.

V tabulce 3.1 jsou zaznamenány výsledky provedených testů. Použitá zkratka HQ značí sadu nahrávek pořízených kvalitnějším mikrofonem a LQ naopak méně kvalitní záznamy. Využití adaptace jazykového kontextu bylo nejprve testováno bez ovlivnění relativní preference poskytnutých výrazů (hodnoty Boost), poté byl tento parametr nastaven na hodnotu 5. Pro normalizaci porovnávaných přepisů a vypočtení hodnoty WER byla použita knihovna JiWER [31].

V disertační práci [32] byly tytéž databáze využity k vyhodnocení úspěšnosti standardního DNN-HMM rozpoznávače s bigramovým jazykovým modelem realizovaného pomocí nástrojové sady Kaldi. Převzaté výsledky jsou pro porovnání zahrnuty ve spodní části tabulky.

Data	Adaptace kontextu	Boost	WER [%]
CtuTest HQ	Ne	-	10,82
CtuTest LQ	Ne	-	10,79
CzLecDSP HQ	Ne	-	16,23
	Ano	-	16,04
	Ano	5	17,75
CzLecDSP LQ	Ne	-	17,13
	Ano	-	17,36
	Ano	5	18,83
CtuTest HQ (převzato [32])	-	-	*15,20*
CzLecDSP HQ (převzato [32])	-	-	*37,40*

Tabulka 3.1: Výsledky provedených testů kvality rozpoznávání

Velmi dobrých výsledků bylo dle očekávání dosaženo při přepisu běžných čtených promluv z databáze CtuTest, kde vypočtený WER nepřesáhl hodnotu 11%. U záznamů spontánních přednesů s odbornou tématikou byla zaznamenána vyšší chybovost, která je však stále na přijatelné úrovni. Použití adaptace jazykového kontextu se slovníkem vygenerovaným přímo z testovaných dat překvapivě zlepšilo

výsledky rozpoznávání pouze u kvalitnějších nahrávek. Zvýšení relativní preference výrazů ze slovníku vedlo v obou případech dokonce k výraznému zhoršení.

Ve srovnání se standardním DNN-HMM systémem dosáhlo použité cloudové řešení výrazně lepších výsledků. Celkově lze kvalitu rozpoznávání hodnotit jako velmi uspokojivou a zcela dostačující pro účel usnadnění orientace v obsahu mediálního záznamu. Kromě kvality rozpoznávání byl testován také čas potřebný k vygenerování přepisu, který vždy odpovídal přibližně jedné čtvrtině délky původního záznamu.

Kapitola 4

Aplikace MediaTranscriber

Hlavním cílem praktické části této práce bylo vytvoření aplikace, která usnadní dohledávání informací v zaznamenaných on-line přednáškách. Tato kapitola je věnována jejímu představení. První sekce obsahuje postup vytvoření aplikace, popis použitých knihoven a shrnutí výzev, kterým bylo během jejího vývoje nutné čelit, a kompromisů, které bylo nutné učinit. Druhá sekce prezentuje funkcionality výsledné aplikace a v závěru kapitoly jsou popsány možnosti jejího využití.

4.1 Vývoj a struktura aplikace

Pro realizaci grafického uživatelského prostředí aplikace byla použita knihovna PyQt [33], která v jazyce Python zprostředkovává funkcionality vývojového frameworku Qt. Ten je hojně využíván pro tvorbu aplikací v jazyce C++ a obsahuje implementaci řady funkčních bloků, ze kterých je možné poskládat kompletní uživatelské prostředí. Použití této knihovny výrazně usnadnilo tvorbu celé aplikace, neboť umožnilo snadno implementovat funkční mediální přehrávač a textový editor, které jsou jejími hlavními součástmi. Rovněž bylo možné využít automaticky generované dialogy pro otevírání a ukládání souborů. Implementační detaily týkající se použití knihovny PyQt nejsou v této práci prezentovány, ale je možné se s nimi seznámit nahlédnutím do zdrojového kódu aplikace, který je uložen na přiloženém CD (Příloha B).

Primární funkce vytvořené aplikace, tedy převod obsahu mediálního souboru do textové podoby, je implementována v modulu transcriber.py. Jeho obsah je do velké míry shodný s obsahem skriptu transcribe_media.py zmiňovaného v sekci 3.3. Rozdíl je především v použití prvků knihovny PyQt k propojení rozpoznávače s ostatními částmi aplikace. Také je zde navíc obsažena funkce pro načtení výsledku rozpoznávání ze souboru ve formátu JSON, která při pouhém přepisu mediálního souboru nebyla zapotřebí. Hlavním nositelem funkcí pro rozpoznávání řeči je třída *Transciber*, jejíž instance umožňují ostatním modulům rozpoznávání realizovat. Předpokladem pro fungování rozpoznávače je přítomnost souborů obsahujících autentizační údaje pro služby Google Cloud a název schránky cloudového úložiště, kam budou nahrávány extrahované zvukové stopy. Při startu aplikace je existence a validita těchto souborů automaticky otestována a v případě problému je funkce rozpoznávání řeči automaticky zablokována. Ostatní funkce aplikace však zůstávají zachovány.

Modul settings.py zajišťuje správu nastavení týkajících se vzhledu aplikace a parametrů rozpoznávání řeči. K tomuto účelu byla vytvořena třída *SettingsHan-*

dler, která obsahuje také samostatné dialogové okno sloužící k úpravám nastavených parametrů. S využitím integrovaných funkcí knihovny PyQt lze zajistit uložení nastavených hodnot do interních registrů operačního systému, a tak zajistit jejich zachování i po restartu aplikace.

Vzhled a funkčnost hlavního aplikačního okna zajišťuje modul gui.py. Jeho klíčovou součástí je třída *MainWindow*, která kromě samotného vzhledu a funkcí grafického prostředí spravuje také všechny ostatní komponenty aplikace. Její součástí jsou tedy objekty typu *Transcriber* a *SettingsHandler*, ale také instance tříd implementovaných uvnitř tohoto modulu. Mezi ně patří třída *WordBrowser* zajišťující funkci prohlížeče výsledků rozpoznávání a s ní související třída *WordEdit*, která realizuje samostatná dialogová okna pro úpravu jednotlivých slov. Třída *ProgressWindow* pak zajišťuje sledování průběhu rozpoznávání řeči ve vyskakovacím okně. V zájmu zajištění funkčnosti uživatelského prostředí v průběhu rozpoznávání je vytvořená instance třídy *Transcriber* přesunuta do samostatného vlákna a její komunikace se zbytkem aplikace je realizována prostřednictvím mechanismu signal-slot, který je implementován knihovnou PyQt.

S využitím výše zmíněných modulů a skriptu main.py, který umožňuje samotné spuštění aplikace, bylo již možné vytvořit funkční nástroj pro přepis obsahu mediálních souborů a jeho následné prohlížení. U každého slova bylo možné zobrazit čas kdy v záznamu zaznělo a pomocí vyskakovacího dialogového okna upravit jeho přepis. Po odladění těchto funkcí byl doplněn modul player.py, který obsahuje kompletní implementaci mediálního přehrávače. Vytvořenou instanci třídy *MediaPlayer* bylo možné snadno zakomponovat do již připraveného hlavního okna aplikace a doplněním funkce pro spuštění přehrávání v čase odpovídajícím vybranému slovu bylo dosaženo stanoveného cíle ohledně fungování aplikace.

Hlavním implementačním problémem, který vyvstal během tvorby aplikace, bylo zajištění přístupu ke kompletním informacím o jednotlivých slovech aniž by je bylo nutné zobrazovat v samotném textu přepisu. Řešením se ukázalo být reprezentování jednotlivých slov pomocí HTML odkazů, které v sobě dokáží uchovat skrytou textovou informaci. Uvnitř nich je vždy uložena dvojice čísel, která jasně definuje pozici daného slova ve struktuře objektu s výstupem rozpoznávače. Při interakci s odkazem jsou pak tyto indexy použity pro vyhledání odpovídajícího dílčího výsledku rozpoznávání a v něm se nacházejícího objektu s informacemi o vybraném slovu. Nevýhodou tohoto přístupu je nutnost generovat zobrazovaný přepis v cyklu pro jednotlivá slova, což může v případě delších záznamů způsobit zamrznutí uživatelského prostředí po dobu až několika sekund. V průběhu tohoto čekání se proto přes hlavní okno aplikace zobrazí informativní grafika signalizující načítání. Další nevýhodou je nemožnost dynamicky měnit vzhled HTML odkazů. Při změně barevného motivu aplikace je proto nutné celý zobrazovaný přepis vygenerovat znovu, aby byl na novém pozadí dobře čitelný. Výhodou generování přepisu po jednotlivých slovech je naopak možnost lépe kontrolovat členění textu do odstavců. Dílčí výsledky rozpoznávání na sebe totiž často navazují uprostřed vět a proto je vhodnější členění realizovat pomocí detekce symbolu tečky na konci jednotlivých slov.

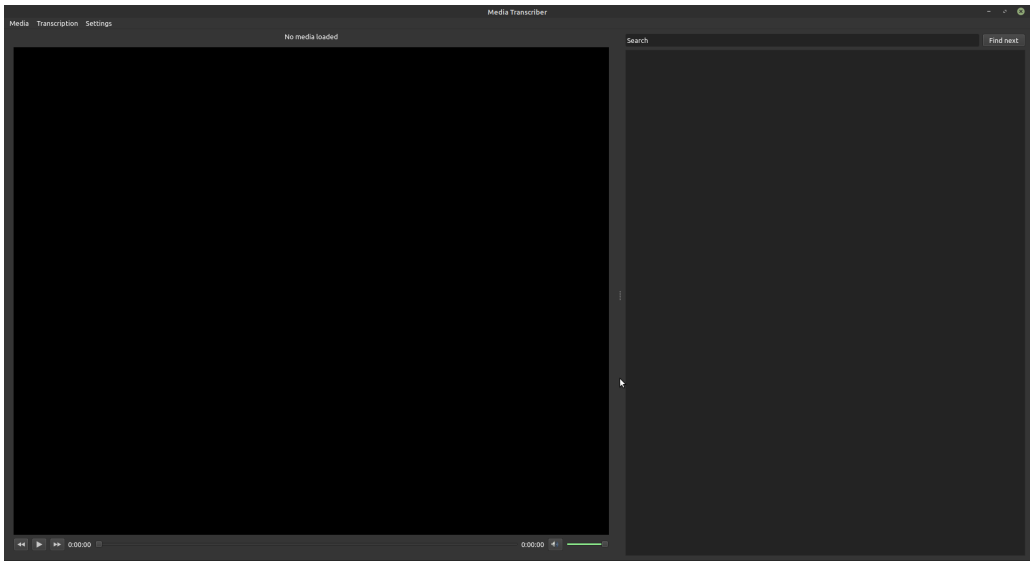
4.2 Použití a funkce

Kompaktní disk, který je součástí příloh této práce obsahuje distribuční složky výsledné aplikace pro linuxové operační systémy a platformu Windows. Obě tyto

složky obsahují binární spustitelný soubor určený pro cílovou platformu, který byl vytvořen s použitím nástroje PyInstaller [34]. Součástí souboru je interpret jazyka Python a kopie všech potřebných knihoven. Díky tomu lze aplikaci používat bez jakýchkoliv zkušeností s vývojem v jazyce Python. Pro spuštění její linuxové verze je však zapotřebí systém obsahující knihovnu glibc ve verzi 2.29 nebo novější. Pro přehrávání některých formátů mediálních souborů může být také třeba doinstalovat potřebné kodeky. V linuxových operačních systémech je k tomuto účelu vhodné využít balíček GStreamer [35], pro systém Windows je pak k dispozici balíček K-Lite Basic [36]. Aplikaci je také možné spustit přímo ze zdrojového kódu, který je rovněž součástí přiloženého CD. K tomu je zapotřebí lokální interpret jazyka Python 3 s doinstalovanými knihovnami, jejichž seznam je uložen v souboru python_requirements.txt přiloženém ke zdrojovému kódu.

Pro využití funkce rozpoznávání řeči musí navíc složka API_config umístěná ve stromovém adresáři aplikace obsahovat platný autentizační soubor platformy Google Cloud nazvaný credentials.json a textový soubor bucket_name.txt obsahující název schránky cloudového úložiště určené k nahrávání zpracovávaných zvukových stop. Během spouštění aplikace proběhne automatická kontrola těchto souborů a v případě neúspěchu se zobrazí chybová hláška s konkrétním problémem. Až do restartu aplikace pak nepůjde vytvářet nové přepisy mediálních souborů. Chyba autentizace může být vyvolána také přerušením připojení k internetu během spouštění aplikace.

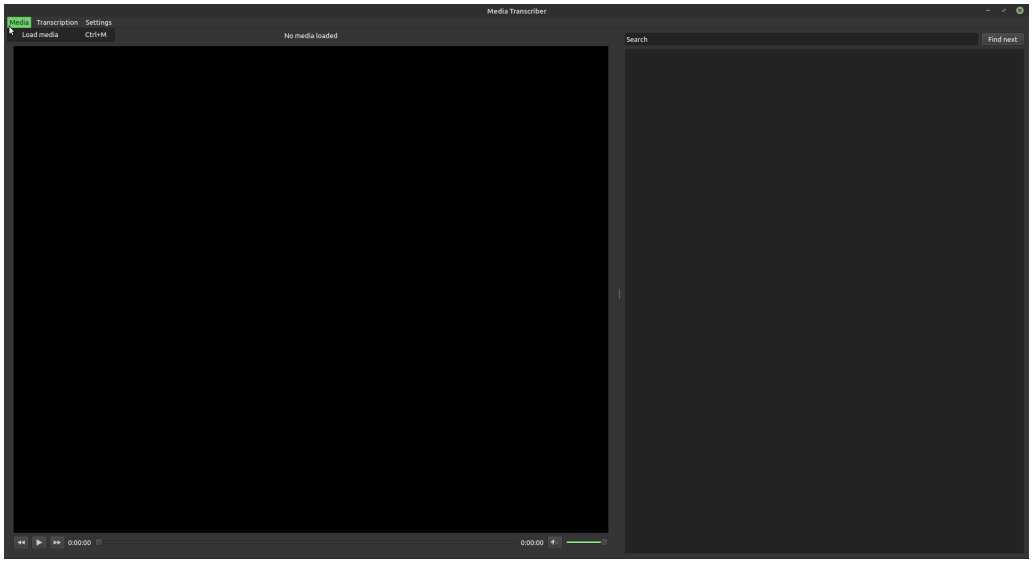
V případě splnění základních prerekvizit se po spuštění aplikace zobrazí hlavní okno uživatelského prostředí zachycené na obrázku 4.1. Jeho hlavní část je rozdělena mezi prvky mediálního přehrávače a textového prohlížeče, jejichž poměr velikostí lze upravovat posouváním dělící hranice. V případě potřeby je možné jednu z těchto částí zcela skrýt a celou plochu hlavního okna tak využít pro jeden účel.



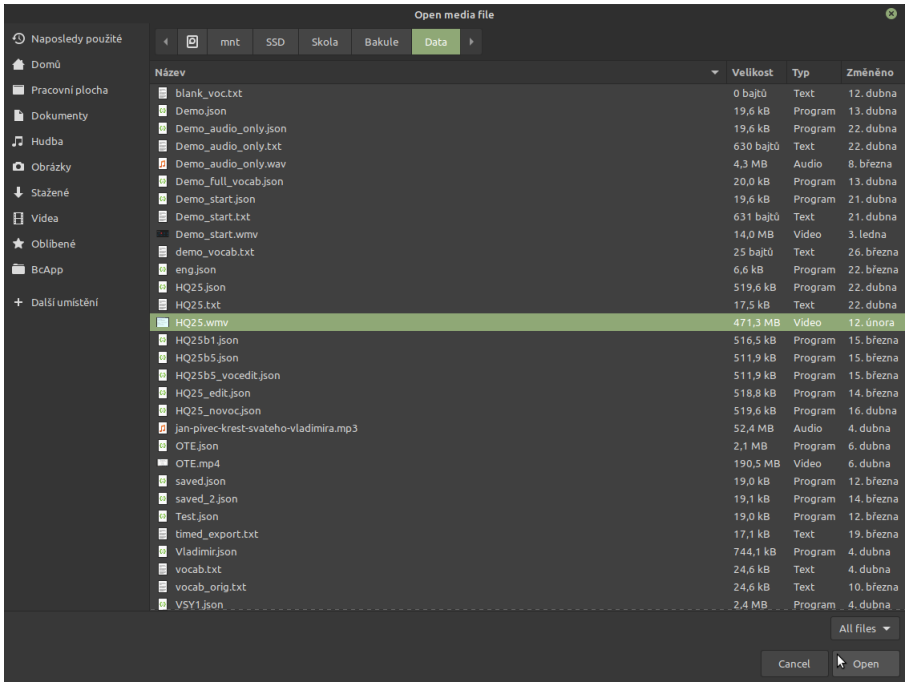
Obrázek 4.1: Hlavní okno aplikace

Načtení mediálního souboru lze provést prostřednictvím záložky „Media“ umístěné v levé horní části hlavního okna (obrázek 4.2). Po kliknutí na jedinou přítomnou položku „Load media“ se otevře dialogové okno pro výběr souboru z úložiště počítače (obrázek 4.3). Po zvolení souboru a potvrzení dialogu dojde k automatické kontrole typu souboru na základě jeho přípony, a v případě, že se nejedná o video nebo zvukovou stopu, se zobrazí chybová hláška. Pokud kontrola proběhne v po-

řádku, automaticky se aktivují ovládací prvky mediálního přehrávače. Mezi ně patří tlačítko pro spuštění a pozastavení přehrávání, tlačítka provádějící třicetisekundové skoky dopředu a zpět v záznamu, posuvník pro přetáčení, tlačítko pro ztlumení zvuku, a posuvník ovládající hlasitost. Stejně jako u většiny mediálních přehrávačů lze i zde využít pro ovládání také klávesu mezerník a čtveřici směrových kláves. V případě, že se zvolený soubor nedaří přehrát, je vhodné zvážit instalaci některého z výše uvedených kodekových balíčků.



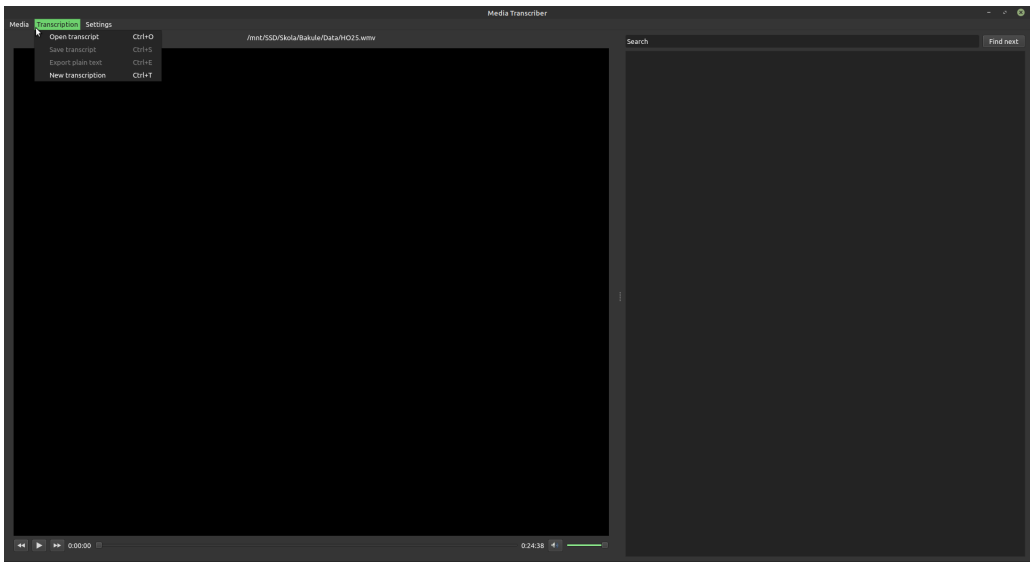
Obrázek 4.2: Obsah záložky Media



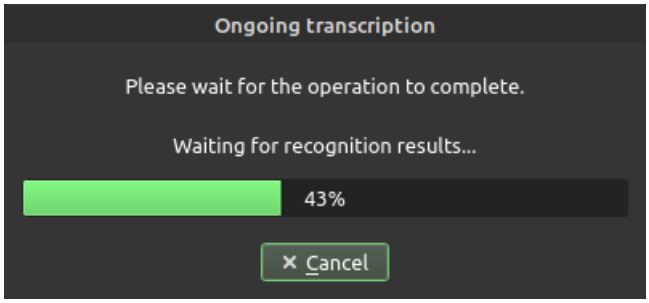
Obrázek 4.3: Dialogové okno pro otevření mediálního souboru

Přepis záznamu lze zahájit kliknutím na položku „New transciption“ uvnitř záložky „Transcription“ (obrázek 4.4). Tímto se otevře nové okno informující o průběhu rozpoznávání (obrázek 4.5). Nejprve je třeba vyčkat na extrakci zvukové stopy

a její nahrání do cloudového úložiště. Průběh samotného rozpoznávání řeči je pak doplněn procentuální a grafickou indikací. Akci pro přepis záznamu lze v kterémkoliv okamžiku zrušit kliknutím na tlačítko „Cancel“. V případě, že k tomu dojde během extrakce zvukové stopy nebo jejího nahrávání je však poté třeba vyčkat než se tato dílčí akce dokončí.

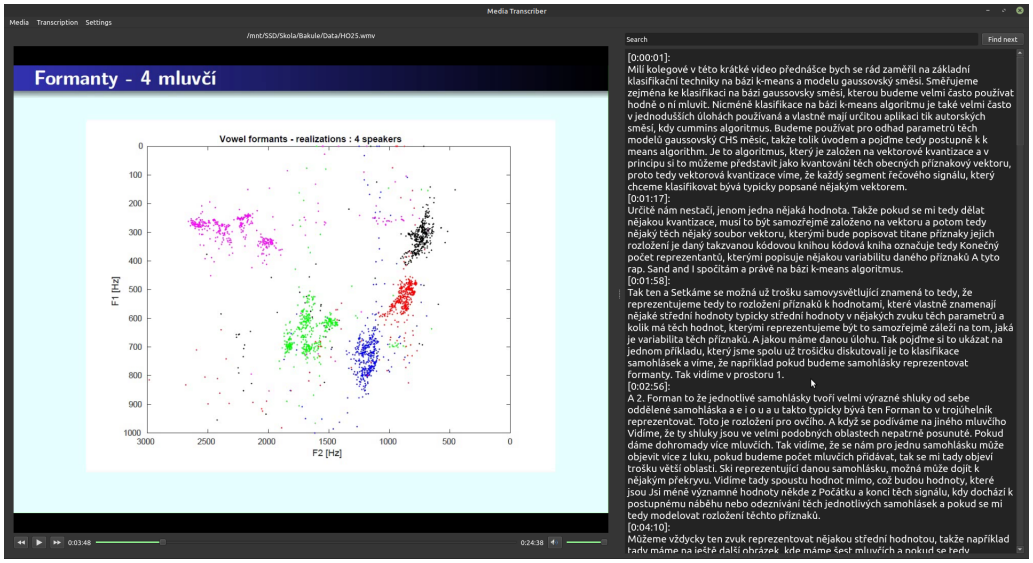


Obrázek 4.4: Obsah záložky Transcription

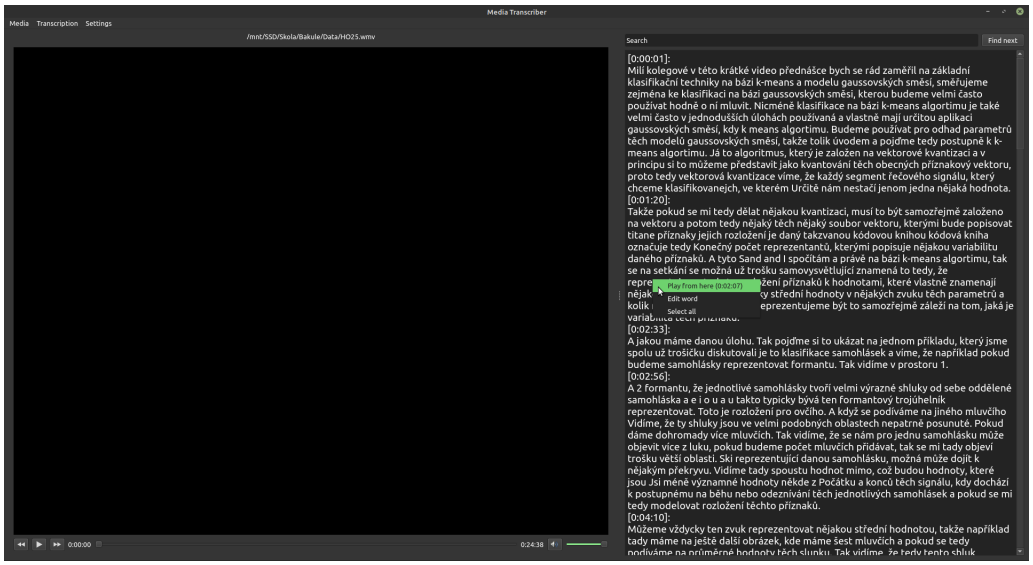


Obrázek 4.5: Okno pro sledování průběhu rozpoznávání řeči

Po úspěšném dokončení procesu rozpoznávání se přepis záznamu automaticky zobrazí v integrovaném textovém prohlížeči (obrázek 4.6). Zobrazený text je členěn do odstavců, které odpovídají přibližně minutovým úsekům původního záznamu. Každý z těchto odstavců je uveden časovou značkou odpovídající začátku prvního slova v odstavci. K vyhledávání konkrétních výrazů uvnitř zobrazeného textu lze využít vyhledávací pole umístěné nad textovým prohlížečem. Opakovaným použitím tlačítka „Find next“ nebo tisknutím klávesy Enter lze postupně procházet jednotlivé shody s hledaným výrazem. Po kliknutí pravým tlačítkem myši na libovolné slovo se zobrazí kontextové menu, jehož první položka „Play from here“ obsahuje v závorce časový údaj o začátku tohoto slova v záznamu (obrázek 4.7). Po kliknutí na tuto položku se mediální přehrávač automaticky přesune do daného okamžiku a spustí přehrávání. Přepis jednotlivých slov lze upravovat výběrem položky „Edit word“ ve zmíněném kontextovém menu, nebo dvojitým kliknutím na vybrané slovo. Podoba okna pro úpravu přepisů slov je zachycena na obrázku 4.8.



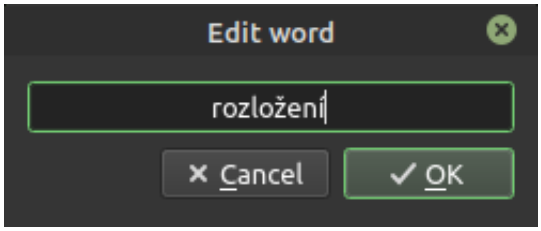
Obrázek 4.6: Ukázka zobrazení přepisu přehrávaného záznamu



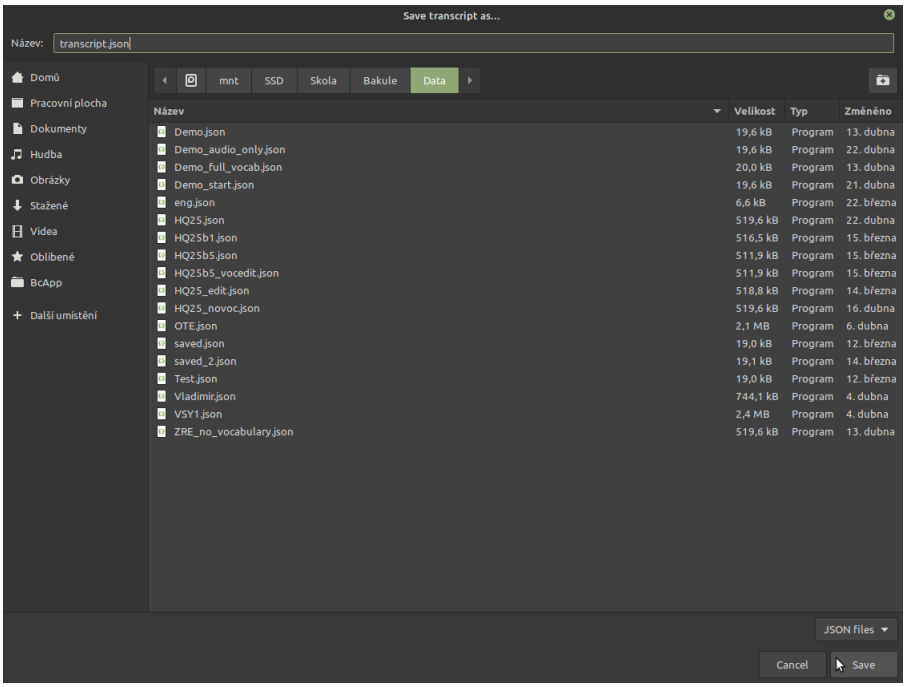
Obrázek 4.7: Kontextové menu jednotlivých slov

Kompletní data vytvořeného přepisu včetně provedených úprav lze uložit do souboru typu JSON prostřednicvím položky „Save transcript“ v záložce „Transcription“, po jejímž zvolení se zobrazí dialog pro uložení souboru (obrázek 4.9). V případě vytváření nového souboru je třeba jeho název zadat včetně přípony „.json“. Tento soubor lze později využít pro opětovné načtení přepisu prostřednictvím volby „Load transcript“. Přepis lze rovněž uložit v čistě textové podobě odpovídající jeho zobrazení v integrovaném textovém prohlížeči. K tomu slouží položka „Export plain text“. Ve vyvolaném dialogovém okně je opět zapotřebí zadat název vytvářeného souboru včetně přípony „.txt“. Takto vytvořené soubory mají stejnou podobu jako výstupy samostatného nástroje pro přepis mediálních záznamů zmíněného v závěru předchozí kapitoly. Ten tak slouží jako alternativa k této aplikaci nevyžadující interakci uživatele s grafickým prostředím.

Po kliknutí na záložku „Settings“ v horní části hlavního okna se zobrazí samostatné okno umožňující správu nastavení aplikace (obrázek 4.10). V sekci pro úpravu



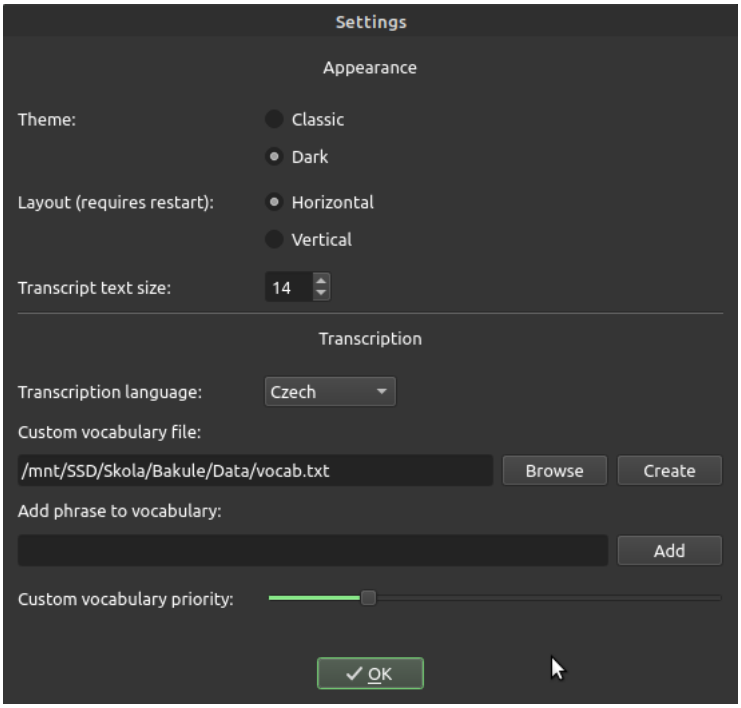
Obrázek 4.8: Dialog pro úpravu přepisu slov



Obrázek 4.9: Dialogové okno pro uložení dat přepisu

vzhledu lze zvolit mezi světlým a tmavým motivem uživatelského prostředí, upravit velikost textu zobrazovaného přepisu a měnit vzájemnou polohu integrovaného mediálního přehrávače a textového prohlížeče. V sekci s předvolbami rozpoznávání řeči lze načíst nebo vytvořit soubor obsahující seznam očekávaných výrazů. Položky tohoto seznamu lze přidávat prostřednictvím textového zadávacího pole. Při pokusu o přidání výrazu, který je v seznamu již obsažen je uživatel automaticky upozorněn. Úspěšné přidání nového výrazu je signalizováno automatickým vymazáním obsahu zadávacího pole. Posuvník umístěný v dolní části dialogového okna slouží k nastavení relativní preference výrazů ze zmíněného seznamu vůči podobně znějícím alternativám. Většina provedených změn má okamžitý efekt, v případě změny rozložení prvků hlavního okna je však k jejímu projevení třeba aplikaci restartovat.

Zmiňované snímky obrazovky byly pořízeny v operačním systému Linux Mint při zvolení tmavého motivu systému i aplikace. V jiných systémech se vzhled aplikačních oken a zejména dialogů pro otevírání a ukládání souborů může lišit.



Obrázek 4.10: Okno pro úpravu nastavení aplikace

4.3 Distribuce a využití

Kromě toho, že jsou distribuční soubory vytvořené aplikace součástí přiloženého CD, jsou rovněž dostupné na webových stránkách Laboratoře zpracování řeči v sekci Ke stažení [37]. Studenti předmětu Zpracování řeči mohou aplikaci použít k procházení přednášek, jejichž přepisy byly vytvořeny v rámci této práce a jsou dostupné v Moodle sekci tohoto předmětu.

Využití nejspíše najdou zejména čistě textové přepisy, které je možné procházet přímo ve webovém prohlížeči. Díky časovému kótování jednotlivých sekcí přepisu je lze využít k orientaci v obsahu přednášky i v případě, že je záznam přehráván z on-line zdroje. Pro využití funkce automatického přetáčení záznamu na začátek vybraného slova je již zapotřebí aby si uživatel záznam stáhl a otevřel jej prostřednictvím dodané aplikace spolu s příslušným přepisem ve formátu JSON.

Kapitola 5

Závěr

V rámci seznámení se s problematikou automatického rozpoznávání řeči byl vytvořen stručný popis principů využívaných při realizaci ASR a byly porovnány základní vlastnosti GMM-HMM, DNN-HMM a End-to-End systémů. Dále byla provedena rešerše dostupných internetových modulů pro automatické rozpoznávání řeči se zaměřením na porovnání jejich funkcí a finančních podmínek jejich použití.

Z dostupných možností byl vybrán modul Google Cloud Speech-to-Text API, byl popsán postup aktivace potřebných služeb platformy Google Cloud a způsob implementace rozpoznávání řeči s využitím tohoto modulu v programovacím jazyce Python. Dále bylo provedeno zhodnocení dosažené kvality rozpoznávání. U čtených obecných promluv z databáze CtuTest byla naměřena nejlepší hodnota WER 10,79%. U záznamů technicky zaměřených přednášek z databáze CzLecDSP bylo dosaženo hodnoty WER 16,23%. Testovaná adaptace jazykového kontextu s využitím slovníku vygenerovaného z databáze CzLecDSP překvapivě ve většině případů vedla ke zhoršení kvality rozpoznávání.

Modul Google Cloud Speech-to-Text API byl dále použit k vytvoření aplikace realizující přepis obsahu multimediálních záznamů. Mezi její funkce, popsané v sekci 4.2, patří také možnost ve vytvořeném přepisu vyhledávat textové výrazy, nebo záznam v integrovaném mediálním přehrávači automaticky přetočit na začátek vybrané promluvy. Aplikace byla použita k vytvoření přepisů přednášek z předmětu Zpracování řeči, které jsou spolu s distribucí samotné aplikace dostupné ze zdrojů popsaných v sekci 4.3. Distribuční soubory včetně zdrojového kódu jsou rovněž součástí přiloženého CD.

V práci byly dosaženy všechny stanovené cíle. Aplikace poskytuje požadované funkce a funguje uspokojivě. Její nevýhodou zůstává nutnost umístění zdrojového záznamu v lokálním úložišti uživatele a závislost na placené službě externího poskytovatele. Předmětem navazující činnosti tak může být přesunutí aplikace do webového prostředí nebo realizace vlastního integrovaného systému pro přepis řeči do textu.

Bibliografie

1. JUANG, B. H.; RABINER, L. R. *Automatic Speech Recognition – A Brief History of the Technology Development* [online]. 2004 [cit. 2021-01-08]. Dostupné z: `https://web.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/354_LALI-ASRHistory-final-10-8.pdf`.
2. IBM CLOUD EDUCATION. *Speech Recognition* [online]. 2020 [cit. 2021-01-07]. Dostupné z: `https://www.ibm.com/cloud/learn/speech-recognition`.
3. LEVIATHAN, Y.; MATIAS, Y. *Google Duplex: An AI System for Accomplishing Real-World Tasks Over the Phone* [online]. 2018 [cit. 2021-01-10]. Dostupné z: `https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html`.
4. DAVIS, K. H.; BIDDULPH, R.; BALASHEK, S. Automatic Recognition of Spoken Digits. *The Journal of the Acoustical Society of America*. 1952, roč. 24, č. 6, s. 637–642. Dostupné z DOI: `10.1121/1.1906946`.
5. *Pioneering Speech Recognition* [online] [cit. 2021-01-09]. Dostupné z: `https://www.ibm.com/ibm/history/ibm100/us/en/icons/speechreco/`.
6. YOUNG, S. The HTK Hidden Markov Model Toolkit: Design and Philosophy. *Entropic Cambridge Research Laboratory, Ltd.* 1994, roč. 2, s. 2–44.
7. UHLÍŘ, J. *Technologie hlasových komunikací*. Vyd. 1. Praha: Nakladatelství ČVUT, 2007. ISBN 978-80-01-03888-8.
8. ŽÁKOVÁ, K. *Implementace rozpoznávače na bázi GMM-HMM v programovém systému MATLAB* [online]. 2020 [cit. 2021-04-17]. Dostupné z: `http://hdl.handle.net/10467/87781`.
9. YU, D.; DENG, L. *Automatic Speech Recognition: A Deep Learning Approach*. 2015. vyd. London: Springer London, 2015. ISBN 978-1-4471-5778-6. Dostupné z DOI: `10.1007/978-1-4471-5779-3`.
10. *CMU Sphinx* [online] [cit. 2021-05-06]. Dostupné z: `https://cmusphinx.github.io`.
11. *About the Kaldi project* [online] [cit. 2021-05-06]. Dostupné z: `http://kaldi-asr.org/doc/about.html`.
12. *Deep Speech Documentation* [online] [cit. 2021-05-06]. Dostupné z: `https://deepspeech.readthedocs.io/en/r0.9/`.
13. HANNUN, A. Y.; CASE, C.; CASPER, J.; CATANZARO, B.; DIAMOS, G.; ELSEN, E.; PRENGER, R.; SATHEESH, S.; SENGUPTA, S.; COATES, A.; NG, A. Y. Deep Speech: Scaling up end-to-end speech recognition. *CoRR*. 2014, roč. abs/1412.5567. Dostupné z arXiv: `1412.5567`.

14. *About Watson Speech to Text* [online] [cit. 2021-05-08]. Dostupné z: https://cloud.ibm.com/docs/speech-to-text?topic=speech-to-text-about#about.

15. *Watson Speech to Text* [online] [cit. 2021-05-08]. Dostupné z: https://www.ibm.com/cloud/watson-speech-to-text.

16. *Language and voice support for the Speech service* [online]. 2021 [cit. 2021-05-07]. Dostupné z: https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/language-support.

17. *Speech to Text* [online] [cit. 2021-05-07]. Dostupné z: https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/.

18. *Speech-to-Text* [online] [cit. 2021-01-14]. Dostupné z: https://cloud.google.com/speech-to-text.

19. *Speech-to-Text basics* [online] [cit. 2021-04-24]. Dostupné z: https://cloud.google.com/speech-to-text/docs/languages.

20. *Amazon Transcribe Features* [online] [cit. 2021-05-08]. Dostupné z: https://aws.amazon.com/transcribe/features/.

21. *What is Amazon Transcribe?* [Online] [cit. 2021-05-08]. Dostupné z: https://docs.aws.amazon.com/transcribe/latest/dg/what-is-transcribe.html.

22. *Getting Started With Wit.ai* [online] [cit. 2021-05-08]. Dostupné z: https://wit.ai/docs.

23. *Wit.ai Frequently Asked Questions* [online] [cit. 2021-05-08]. Dostupné z: https://wit.ai/faq.

24. ZHANG, A. *SpeechRecognition* [online]. 2017 [cit. 2021-01-11]. Dostupné z: https://pypi.org/project/SpeechRecognition/.

25. PHAM, H. *PyAudio* [online]. 2017 [cit. 2021-04-25]. Dostupné z: http://people.csail.mit.edu/hubert/pyaudio/#downloads.

26. *Speech-to-Text basics* [online] [cit. 2021-04-24]. Dostupné z: https://cloud.google.com/speech-to-text/docs/basics.

27. *Google Cloud* [online] [cit. 2021-05-09]. Dostupné z: https://cloud.google.com.

28. GOOGLE LLC. *google-cloud-speech* [online]. 2021 [cit. 2021-04-25]. Dostupné z: https://pypi.org/project/google-cloud-speech/.

29. GOOGLE LLC. *google-cloud-storage* [online]. 2021 [cit. 2021-04-25]. Dostupné z: https://pypi.org/project/google-cloud-storage/.

30. *MoviePy* [online]. 2020 [cit. 2021-04-25]. Dostupné z: https://zulko.github.io/moviepy/.

31. VEASSEN, N. *JiWER* [online]. 2020 [cit. 2021-05-16]. Dostupné z: https://pypi.org/project/jiwer/.

32. MIZERA, P. *Applying articulatory features within speech recognition* [online]. 2019 [cit. 2021-05-20]. Dostupné z: http://hdl.handle.net/10467/85527.

33. RIVERBANK COMPUTING LIMITED. *PyQt5* [online]. 2021 [cit. 2021-04-30]. Dostupné z: https://pypi.org/project/PyQt5/.

34. *PyInstaller* [online]. 2021 [cit. 2021-05-01]. Dostupné z: `https://www.pyinstaller.org/index.html`.

35. *GStreamer* [online]. 2021 [cit. 2021-05-01]. Dostupné z: `https://gstreamer.freedesktop.org`.

36. *K-Lite Codec Pack* [online]. 2021 [cit. 2021-05-01]. Dostupné z: `https://codecguide.com/download_kl.htm`.

37. *Laboratoř zpracování řečového signálu, Ke stažení* [online] [cit. 2021-05-20]. Dostupné z: `http://noel.feld.cvut.cz/speechlab/start.php?page=download&lang=cz`.

Přílohy

A Ukázkové kódy

Skript pro přepis lokálního zvukového souboru uvedeného v argumentu volání

```
import speech_recognition as sr
import sys

r = sr.Recognizer()
f = sr.AudioFile(sys.argv[1])

with f as source:
    audio = r.record(source)

recog = r.recognize_google(audio, language="cs")
print(recog)
```

Skript pro přepis řeči zaznamenané mikrofonem

```
import speech_recognition as sr

r = sr.Recognizer()
mic = sr.Microphone()

print("Mluvte:")
with mic as source:
    r.adjust_for_ambient_noise(source) #redukce sumu
    audio = r.listen(source)

print("Probiha rozpoznani...")
recog = r.recognize_google(audio, language="cs")
print(recog)
```

B Obsah přiloženého CD

V kořenovém adresáři kompaktního disku odevzdaného společně s touto prací jsou uloženy:

- PDF soubor obsahující tuto práci v elektronické podobě
- Složka MediaTranscriber obsahující distribuční soubory vytvořené aplikace včetně jejího zdrojového kódu a skriptu pro přímý přepis mediálního záznamu
- Složka Data obsahující ukázkové videozáznamy spolu s příslušnými přepisy