Advanced User Interface – 20/21

Speech-To-Text Module

07.02.2021

—

Aida Gasanova,
aida.gasanova@mail.polimi.it Computer Science and Engineering

Max Griesmayer,
max.griesmayer@mail.polimi.it Erasmus - Computer Science and Engineering

Victor Mouradian
victor.mouradian@gmail.com Erasmus - Computer Science and Engineering

Figure 1 Picture from one oif many online meetings

Abstract

In this project we tried to face the challenge of creating a speech-to-text module that enables the transcription of children with language disorder. In this document we describe how we approached this challenge. We explain our goals, the state of the art and the difficulties we had on our way to a functioning speech to text model. Starting with four people, we lost our only Italian speaking member after 2 months into the project. That kicked us back a little bit but didn't stop us from building new models to achieve our goal. In the end we managed to develop an Italian open-source speech to text module. The goal of developing one for kids with DLD has to be postponed till there is enough data available.

Table of Content

Introduction

Our project is the development of a speech to text module aimed for children having Developmental Language Disorder (DLD) meaning children that have issues with understanding and/or using languages.

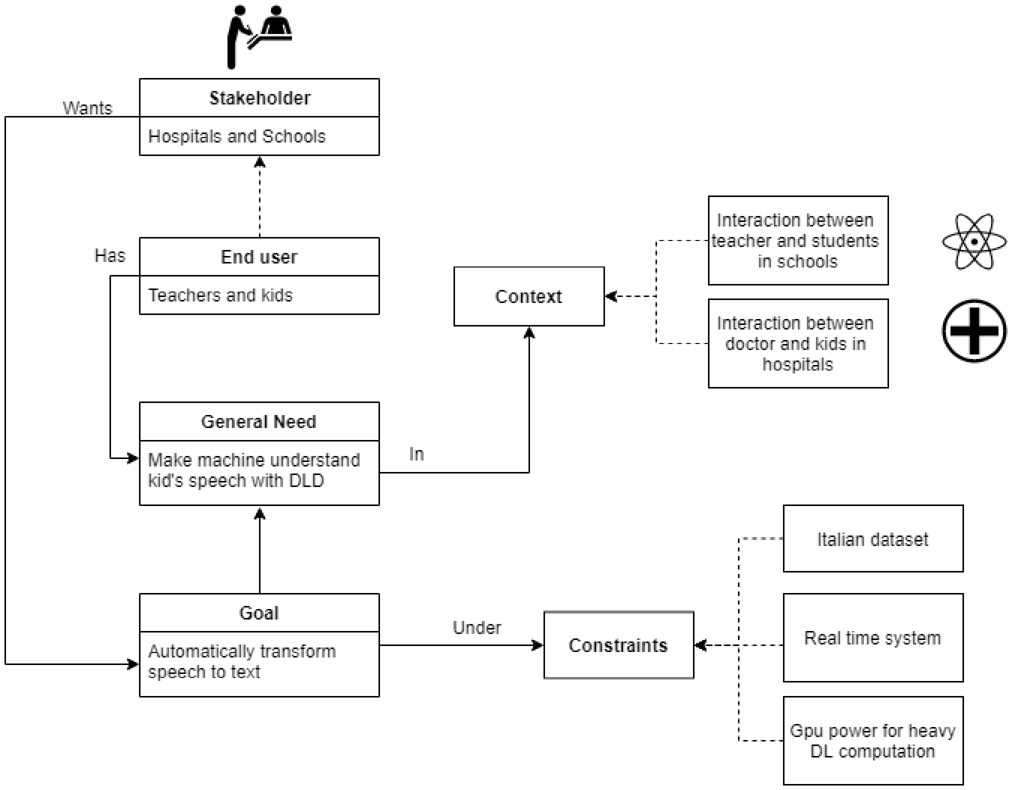

Wants
Stakeholder
Hospitals and Schools
Has
End user
Teachers and kids
General Need
Make machine understand kid's speech with DLD
In
Context
Interaction between teacher and students in schools
Interaction between doctor and kids in hospitals
Goal
Automatically transform speech to text
Under
Constraints
Italian dataset
Real time system
Gpu power for heavy DL computation

Figure 2 goal diagram

Early on we have decided that the best course of action was for us to develop a natural language processing (NLP) module using deep learning technology.
This choice allowed us to envision how we would tackle this problem. The development of the solution was thus categorised into 3 different steps.
But also due to the sheer size of having to develop a speech to text module like this it was convened that we wouldn't be doing a UX for our project and focus on having the speech to text module working.

The first was choosing a base for our project, many speech to text modules are available today (Google, IBM, Mozilla) each one of them with their own success rate but not all of them are available in Italian and open source, thus many didn't fit the requirement for our project.

So our choice ended up being the Deep Speech module made by the Mozilla foundation which is open source and allows us to get a model (STT module here) with the data of our choice.

Which brought us to our data gathering essential in order to have some results, DLD data in Italian was not something widely available on the web, but we got lucky because Micol Spitale and Fabio Catania, our mentors for this project, had some data gathered from working with children with speech impairment. We then had to convert this data into the format that was supported by deepspeech, which we managed to after evaluating what the format required and how we could go from what we add to what we want.

The last step was training our model on the DLD data and then evaluating our model to other solutions available. This step brought us many problems because the data was not in the right format, we had dependencies errors and many more.

We hope you are going to find our project interesting, as much as we do and that it is just the first stone on having a speech to text module for kids with learning disabilities as it is an important subject. We have added directions into the future work sections, and please don’t hesitate to contact any members of the team for help in the future.

Target Groups and User Needs

The Goal of our project is to develop a Speech-to-Text (STT) module which is able to understand the speech of children with Developmental Language Disorders (DLD). In this project we only focus on the italian language.

Our main target groups are the children with DLD, the therapists working with the children, and the parents or other relatives of the child.

To better understand our main target group we need to declare what DLD is and what the challenges are. DLD stands for Developmental Language Disorder. Having DLD means that a child or young person has severe, persistent difficulties understanding or using spoken language.
DLD is diagnosed by a Speech and Language Therapist only and is used for children over the age of 5 years. DLD is only identified when a child continues to have severe Language and Communication Needs following targeted intervention.

There is no known cause of DLD which can make it hard to explain. DLD is not caused by other biomedical conditions (such as ASD, hearing loss), emotional difficulties or limited exposure to language.
DLD can co-occur with other difficulties such as Attention Deficit Hyperactivity Disorder (ADHD), dyslexia and speech sound difficulties.

Even though non verbal communication is very important for children with DLD, in this project we only focus on verbal communication.

To reach our goal we want to take an existing italian TTL model and use existing DLD Data to create a tool that is able to understand and transcribe kids with DLD. To make this work we plan to use transfer learning.

The biggest constraint on how good the tool will be that we create, is the quantity and the quality of the existing data. To create a model that is good enough to understand we need a lot of transcribed data in the right format.

Since the persons working with this tool are no technicians, it is an important need to make the technology as easy-to-use and user friendly as possible. To later use our tool, it is necessary that the user can interact with it without frustration.

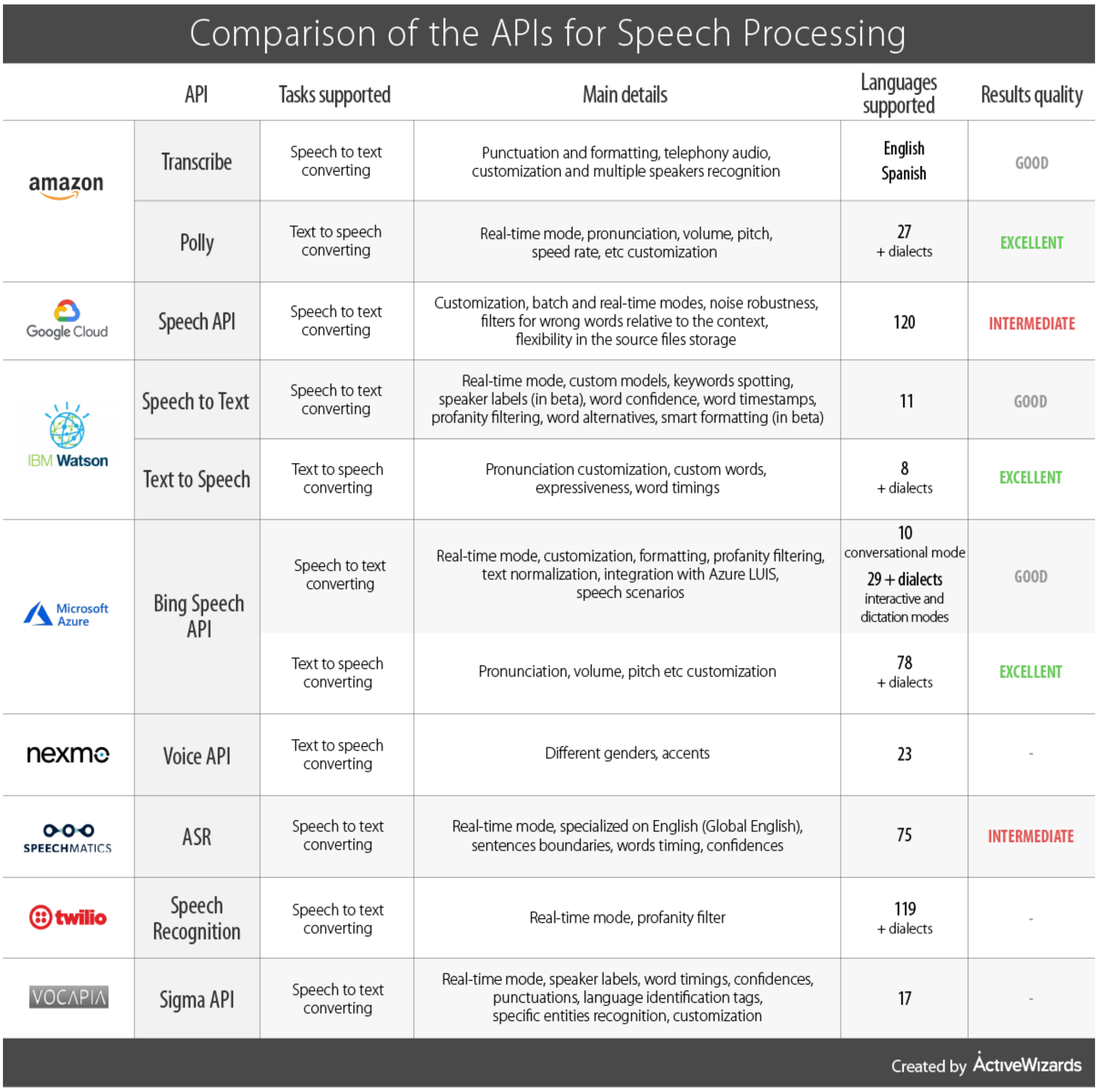
A further user need is privacy. Since the module should be used for a therapy session and its used in medical area. Therefore, the module must be aware of data privacy over the whole pipeline. We are going to build an Open-Source model to ensure full data privacy.

Another advantage of creating an open-source solution is full visibility into the code. That will allow us and future users and collaborators a discussion about the development. Furthermore, it protects against lock-in risks.

A third advantage is the ability to make changes & add additional features to let the project grow over time. So, if there are new approaches and technologies coming up its easy to react and adopt the program. Being the owner of the model assures us that we can deploy model wherever we want and use it offline as well as hosted online in the cloud. It even enables us to create a docker container to make it scalable on a high scala.

State of the art

There are many options for speech processing, some propose transcription (Speech to text) others Text to speech.



Comparison of the APIs for Speech Processing

	API	Tasks supported	Main details	Languages supported	Results quality
amazon	Transcribe	Speech to text converting	Punctuation and formatting, telephony audio, customization and multiple speakers recognition	English Spanish	GOOD
	Polly	Text to speech converting	Real-time mode, pronunciation, volume, pitch, speed rate, etc customization	27 + dialects	EXCELLENT
Google Cloud	Speech API	Speech to text converting	Customization, batch and real-time modes, noise robustness, filters for wrong words relative to the context, flexibility in the source files storage	120	INTERMEDIATE
IBM Watson	Speech to Text	Speech to text converting	Real-time mode, custom models, keywords spotting, speaker labels (in beta), word confidence, word timestamps, profanity filtering, word alternatives, smart formatting (in beta)	11	GOOD
	Text to Speech	Text to speech converting	Pronunciation customization, custom words, expressiveness, word timings	8 + dialects	EXCELLENT
Microsoft Azure	Bing Speech API	Speech to text converting	Real-time mode, customization, formatting, profanity filtering, text normalization, integration with Azure LUIS, speech scenarios	10 conversational mode 29 + dialects interactive and dictation modes	GOOD
		Text to speech converting	Pronunciation, volume, pitch etc customization	78 + dialects	EXCELLENT
nexmo	Voice API	Text to speech converting	Different genders, accents	23	-
SPEECHMATICS	ASR	Speech to text converting	Real-time mode, specialized on English (Global English), sentences boundaries, words timing, confidences	75	INTERMEDIATE
twilio	Speech Recognition	Speech to text converting	Real-time mode, profanity filter	119 + dialects	-
VOCAPIA	Sigma API	Speech to text converting	Real-time mode, speaker labels, word timings, confidences, punctuations, language identification tags, specific entities recognition, customization	17	-

Created by ActiveWizards

Figure 3 Companies selling speech processing solutions

Many of the different solutions are proposed by privately owned companies and thus sell the access to their service. They propose an easy to access solution if you ever want to transcribe some audio by giving you an API to access and process everything on their server.

But due to the nature of our project, and because we wanted a speech to text module suited to our needs, our choice was logically oriented towards open-source solutions because those allow for modification and addition of new data.

Here is a non-extensive list of the best speech recognition models open source and available on github.

DeepSpeech :

- DeepSpeech is an open-source speech to text engine which can run in real-time using a model trained by machine learning techniques based on Baidu's Deep Speech research paper and is implemented using Tensorflow.
- DeepSpeech can run in real-time on devices ranging from a Rasberry Pi 4 to any power GPU Servers and it supports various platforms for its development such as: Linux, Android, Windows and macOS.

Figure 4 Mozilla deepspeech

Leon :

- Leon is an open-source personal assistant who can live on your server and is able to perform task when you ask him to. You can talk to him and he can talk to you, you can text him and he can text you back and the best part is Leon can communicate with you by being offline to protect your privacy.
- Leon is open-source and uses AI concepts. It is built mainly using Node.js and Python and supported operating systems include: Linux, MacOS and Windows.

Figure 5 Leon

Wav2letter :

- Wav2letter++ is Facebook AI Research's end-to-end Automatic Speech Recognition Toolkit written entirely in C++, supporting a wide range of models and learning techniques. It is often compared to DeepSpeech due to the many similarities in the two.
- Wav2letter++ also embarks a very efficient modular beam-search decoder, for both structured learning (CTC, ASG) and seq2seq approaches.Their Github repository includes recipes to reproduce the following research papers as well as pre-trained models.

Figure 6 wav2letter

-

We can also think of Annyang written in javascript and Speech recognition written in python.

We have chosen to go with the DeepSpeech solution, as it is a technology understood by members of the group (Tensorflow) and has an italian version, something essential to our project.

Implementation

DeepSpeech model

As specified in the state of the art, in order to build our Speech To Text module we have decided to use Deep Learning technology and base our model on the work done on the DeepSpeech open-source project.

The model is based on Baidu's Deep Speech research paper and is implemented using Google tensorflow in order to make the implementation easier and thus encouraging the community to contribute to the project. The latest release of the model has achieved 6.5% of Word Error Rate on LibriSpeech's test-clean set (in English).

The model implements a recurrent neural network (RNN) architecture, trained to ingest speech spectrograms and generate English text transcriptions.

"A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs.This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition"

4. Wikipedia definition for RNN

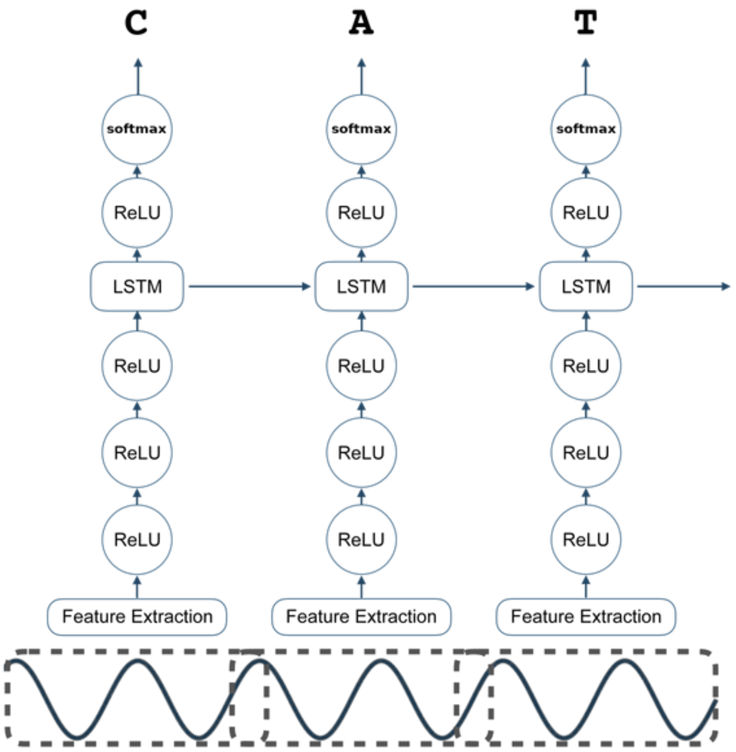


Figure 7: The architecture of the DeepSpeech Model

The original model was based on English language, and is now available in French, German, Spanish, Italian, Chinese…

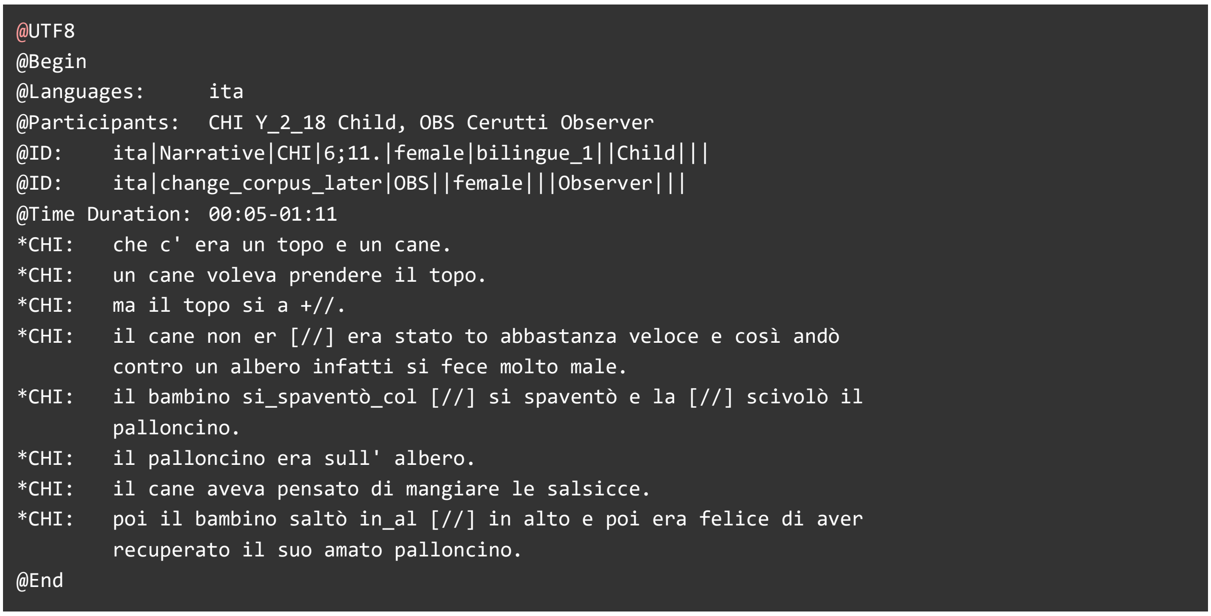
In deep learning, there is a technique called fine-tuning which is useful in bootstrapping the training of a new model based on new data. It is especially great when you don't have access to a lot of data as this is our case with the DLD data given to us.

Which brings us to our next step in our evolution, which is data formatting, essential in order for us to start fine tuning a model using the data.

Data formatting

The data given to us was as followed:

- **LD_audio_ITA** (806mo), which contains 94 audio recordings, organised by year and session. The audio files all have different length, audio quality and actors. The files are in wav and mp3 format, some have a bitrate of 1411 Kbit/s.
- **LD_ITA**, which contains the transcript for each audio file. The transcripts are in a cha format and look like this :



```
@UTF8
@Begin
@Languages:     ita
@Participants:  CHI Y_2_18 Child, OBS Cerutti Observer
@ID:    ita|Narrative|CHI|6;11.|female|bilingue_1||Child|||
@ID:    ita|change_corpus_later|OBS||female|||Observer|||
@Time Duration: 00:05-01:11
*CHI:   che c' era un topo e un cane.
*CHI:   un cane voleva prendere il topo.
*CHI:   ma il topo si a +//.
*CHI:   il cane non er [//] era stato to abbastanza veloce e così andò
        contro un albero infatti si fece molto male.
*CHI:   il bambino si_spaventò_col [//] si spaventò e la [//] scivolò il
        palloncino.
*CHI:   il palloncino era sull' albero.
*CHI:   il cane aveva pensato di mangiare le salsicce.
*CHI:   poi il bambino saltò in_al [//] in alto e poi era felice di aver
        recuperato il suo amato palloncino.
@End
```

Figure 8 transcript

The transcript gives us multiple info, but the most interesting is the [//] symbol as it used each time the child makes a mistake when speaking and thus highlights the speech impairment.
Unfortunately Deep Speech doesn't understand the transcript file, this mean that we have to convert the audio files and their associated transcript files into a format deepspeech can understand.
The format is a csv file with the following fields :

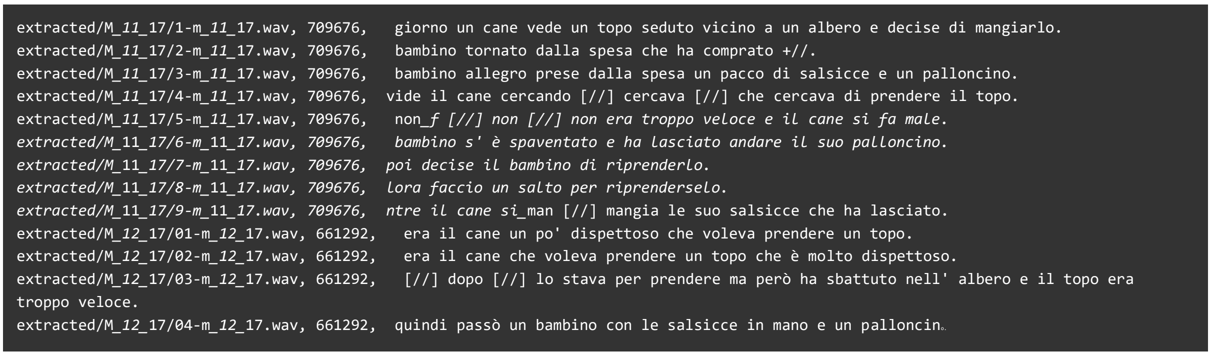
- wav_filename - path of the sample, either absolute or relative. Here, the importer produces relative paths.
- wav_filesize - samples size given in bytes, used for sorting the data before training. Expects integer.
- transcript - transcription target for the sample.

The transcript needs also to be a single phrase for better results.

Because of the sheer size of the audio, the difficulty to extract manually the right audio then write the associated phrase to a csv file and the fact that the only italian speaker in our group left the project; we decided to automate the process using python.

We have written two python scripts to help us convert the DLD data into the right format.

- **extractaudio.py**
  The script will read the transcript associated to each audio file and figure out two things, the number of phrases said, and the total length of the conversation (@Time Duration in the .cha file), to reach an average time for each phrase and then extract each phrase from the audio into an audio file using the average time for each phrase.
  The output is a zip folder containing each one of the extracted audio, each one being in the same folder as the **LD_audio_ITA** folder and the associated csv file with the phrase corresponding to each audio.
  The only issue due to the library used in the script was that the output format for the audio was in mp3.
  Using this script we went from 98 audio files to 848.
- **convertwav.py**
  This script was in charge of converting all mp3 into a single channel wav file.
  It also updates the csv file generated by the **extractaudio.py** file.
  This also allows to reduce the size of the initial dataset from 806mo to 554mo.



```
extracted/M_11_17/1-m_11_17.wav, 709676,   giorno un cane vede un topo seduto vicino a un albero e decise di mangiarlo.
extracted/M_11_17/2-m_11_17.wav, 709676,   bambino tornato dalla spesa che ha comprato +//.
extracted/M_11_17/3-m_11_17.wav, 709676,   bambino allegro prese dalla spesa un pacco di salsicce e un palloncino.
extracted/M_11_17/4-m_11_17.wav, 709676,  vide il cane cercando [//] cercava [//] che cercava di prendere il topo.
extracted/M_11_17/5-m_11_17.wav, 709676,   non_f [//] non [//] non era troppo veloce e il cane si fa male.
extracted/M_11_17/6-m_11_17.wav, 709676,   bambino s' è spaventato e ha lasciato andare il suo palloncino.
extracted/M_11_17/7-m_11_17.wav, 709676,  poi decise il bambino di riprenderlo.
extracted/M_11_17/8-m_11_17.wav, 709676,  lora faccio un salto per riprenderselo.
extracted/M_11_17/9-m_11_17.wav, 709676,  ntre il cane si_man [//] mangia le suo salsicce che ha lasciato.
extracted/M_12_17/01-m_12_17.wav, 661292,   era il cane un po' dispettoso che voleva prendere un topo.
extracted/M_12_17/02-m_12_17.wav, 661292,   era il cane che voleva prendere un topo che è molto dispettoso.
extracted/M_12_17/03-m_12_17.wav, 661292,   [//] dopo [//] lo stava per prendere ma però ha sbattuto nell' albero e il topo era
troppo veloce.
extracted/M_12_17/04-m_12_17.wav, 661292,  quindi passò un bambino con le salsicce in mano e un palloncino
```

Figure 9 An extract of the csv file

Model training

Training environment:

For all the training we used Python using google colab:

Since a GPU is needed to train the model and none of us had one available we used Google Colab for the development and model training.

“Colab” stands for Colaboratory, and it is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. It also enabled us to work remotely and together on the same project.

To train the model we used Tensorflow and mainly the deepspeech library which already supplies a working and proved deep learning network for speech to text training.

Training of an Italian speech model:

First, we tried to build a model from scratch using all the data that was available to us. As explained in section 6 we used publicly available data from the Mozzila Common-Voice dataset and the MAILABS dataset. Due to the fact that we have a total amount of over 250 hours of transcribed audio files. We were reaching the limits of the system. We had to split the huge amount of data that is processed here, each training cycle needed multiple hours and one model would need up to 30 cycles(epochs).

Transfer Learning

Transfer learning is not a new concept. The first time the phrase came up was at the The Neural Information Processing Systems (NIPS) 1995 workshop Learning to Learn: Knowledge Consolidation and Transfer in Inductive Systems.

Thus, the key motivation, especially considering the context of deep learning is the fact that most models which solve complex problems need a whole lot of data, and getting vast amounts of labeled data for supervised models can be really difficult, considering the time and effort it takes to label data points.

Traditional learning is isolated and occurs purely based on specific tasks, datasets and training separate isolated models on them. No knowledge is retained which can be transferred from one model to another. In transfer learning, you can leverage knowledge (features, weights etc) from previously trained models for training newer models and even tackle problems like having less data for the newer task!

Training of the DLD model

After we managed to create an Italian language STT model we focused on the DLD part. Also here we tried different approaches with different parameters

Transfer learning:

With the same approach like stated before we tried to train the model to understand the DLD data we got. It didn’t took us long to realize that we have not enough data and that we need a different approach to reach our goal

Fine tuning

Fine Tuning means taking weights of a trained neural network and using it as initialization for a new model being trained on data from the same domain. It is used to:

- speed up the training
- overcome small dataset size

There are various strategies, such as training the whole initialized network or "freezing" some of the pre-trained weights (usually whole layers). In our case we used it because of the small size of the dataset.

We took our Italian speech to text model and used it as a checkpoint to retrain with the DLD data.

We realized very early that the results were not satisfying. The accuracy looked promising but when we took a closer look at the words and sentences produced, we realized most of them do not make sense. So, we worked on data cleaning with different methods, but we were not able to get satisfying DLD results.

Value Proposition

This project was a way full of challenges. One challenge followed the next. In the end it was a big journey of learning. We had to accept that the problem we faced was way more complicated than we thought in the beginning.

The biggest challenges were:

- data quality
- data size
- data annotation
- computational power
- memory
- file space
- time

In fact, we had the classic problems someone faces when you are working on a data science project. Now we can split the challenges into two different types of problems. The one regarding hardware (like computational power, memory, or file space) and the problems regarding data.

For the first part there are workarounds we had to implement to split the data, save intermediate results locally, and let the script run over multiple days.

The big problem was the data. Our approaches to face these problems:

- Data quality:

  We used different data cleaning approaches to get rid of background noises and make the voice louder. Due to the fact there is too much data we had to write a script that does it automatically.

- Data size:

There are many fields where it is possible to create additional data using augmented data. For example, data augmentation techniques such as cropping, padding, and horizontal flipping for image recognition. The problem here is that these techniques are not usable in the field of STT.

- Data Annotation

The given data was missing the timestamps when a sentence or word is spoken. Since it was too much data to do it manually, we created an automated solution. This worked very well, but of course it could not generate perfect time stamps. Which leads to worse performance of the model

Data gathering tool

To tackle the problem and to be able to create a functional STT module for DLD, it is necessary to gather a lot of high-quality data. To assure that this data has the right format and is labelled the right way using timestamps. Here is a list of needs we figured out for a data gathering tool.

Usability

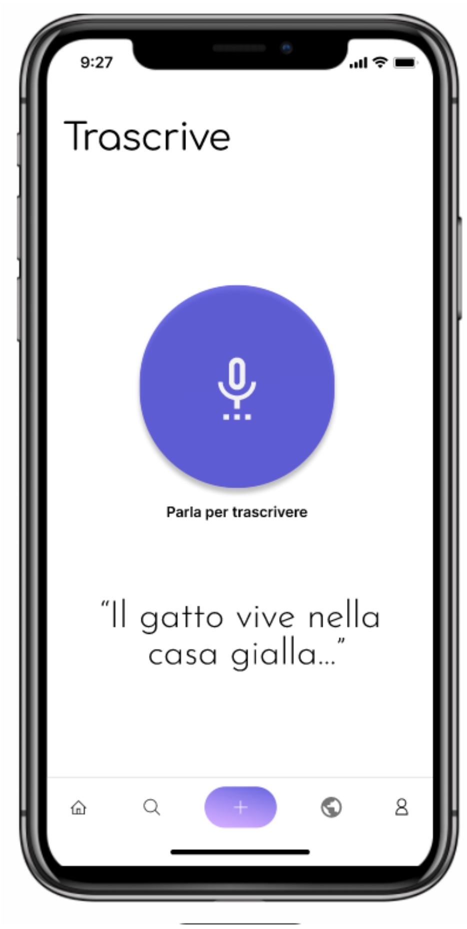
Since the tool will be used by kids, therapists, and their parents its necessary to ensure a self-explanatory design. It should be easy to use.

Gamification

Since this tool will be a crowdsourcing tool where we need the users to be motivated to use the tool. The easiest way to achieve that is to use gamification. Gamification describes the incentivisation of people's engagement in non-game contexts and activities by using game-style mechanics. Gamification leverages people's natural tendencies for competition, achievement, collaboration, and charity.

Annotation

We thought that the easiest way to gather this data would be prewritten sentences the kid has to say. Regarding our problem that would not be that easy since that would limit the therapist's work and conversation with the child. To ensure a free and open conversation it will be necessary for the therapist to transcribe the sentences afterwards. Also, here gamification, like rewards could help to keep the therapist motivated



“Il gatto vive nella casa gialla...”

Figure 10 Data gathering prototype

Bibliography

Hannun, Awni, et al. "Deep speech: Scaling up end-to-end speech recognition." *arXiv preprint arXiv:1412.5567* (2014).

Amodei, Dario, et al. "Deep speech 2: End-to-end speech recognition in english and mandarin." *International conference on machine learning*. PMLR, 2016

Bijl, David, and Henry Hyde-Thomson. "Speech to text conversion." U.S. Patent No. 6,173,259. 9 Jan. 2001.

https://deepspeech.readthedocs.io/en/v0.9.3/
https://github.com/mozilla/DeepSpeech
https://pypi.org/project/SpeechRecognition/
https://github.com/TalAter/annyang
https://www.kdnuggets.com/2018/12/activewizards-comparison-speech-processing-apis.html
https://waliamrinal.medium.com/top-5-speech-recognition-open-source-projects-and-libraries-with-most-stars-on-github-d705408b834

Data:
https://commonvoice.mozilla.org/
https://www.caito.de/2019/01/the-m-ailabs-speech-dataset/

.

List of Figures