Automatic Speech Recognition

Dargier Antoine

Ponchon Martin

06/02/2023

Introduction, Motivation & Related Work

1952
Recognition of 10 digits

1970s
IBM works with Jelinek

1972
First recognition system

2008
Google voice-activated search tool

2011
Apple and Siri

2017
Microsoft announces that it's a solved problem

Methodology

Dataset

French Single Speaker Speech Dataset: 8,600 audios of *Les Misérables* and *Arsène Lupin contre Herlock Sholmes* and their transcripts in French

Metric

$$WER = \frac{S + D + I}{N}$$

where:

- S = number of substitutions (word replaced by another)
- D = number of deletions (word removed)
- I = number of insertions (word added)
- N = number of words in the reference

Traditional ASR pipeline

- Acoustic Model: audio to phonemes

 → HH AH L OW

- Pronunciation Model: phonemes to word

HH AH L OW → Hello

- Language Model: Word to sentence

Hello → 

The CNN model

Pre-processing → Building the train/test sets → Building the CNN model → Validation

Pre-processing

Compute **Mel-frequency cepstral coefficients** (MFCC) features on audios: short term power spectrum of a sound

Building the train/test sets

Build input and output of the model computing the MFCC and converting vectors into binary matrix

Building the CNN model

Conv2D – MaxPooling2D – Conv2D – MaxPooling2D – Flatten – Dense – Dropout – Dense

Validation

8 hours of training on 2,000 audios of Kaggle dataset

Results

Results not usable because the learning time was too limited

WER > 20% in the literature

The CTC model

Characteristics:

- Trained on **labelled data** (i.e. transcribed audio)
- Implemented as **a grapheme based model**, i.e. working with characters directly instead of phonemes
- Solves the issue **of different lengths between input and output transcription** (using blank characters)

Idea:

- Pre-process raw audio to obtain a simple **spectrogram** (FFT of small windows of waveform) to get frequency content
- Use RNN, whose output neurons encode a **distribution over symbols** $c \in \{A, B, \ldots, space, blank\}$
- Define a **mapping** encoding $c \longmapsto y$ word: the transcription is obtained by removing duplicates and blank characters: HHH_EE_LL_LO ⟼ HELLO

Network parameters are updated with **backpropagation** and gradient descent to **maximize likelihood of choosing the correct label**: $\boldsymbol{L}(\boldsymbol{\theta}) = \log \mathbb{P}\left(\boldsymbol{y}^{*(i)} \middle| \boldsymbol{x}^{(i)}\right) = \boldsymbol{CTC}\left(\boldsymbol{c}^{(i)}, \boldsymbol{y}^{*(i)}\right)$

Results

Results not usable because insufficient processing power

WER = 16% in the literature

Wav2Vec2.0

Developed by Meta in 2019

Authors: Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, Michael Auli

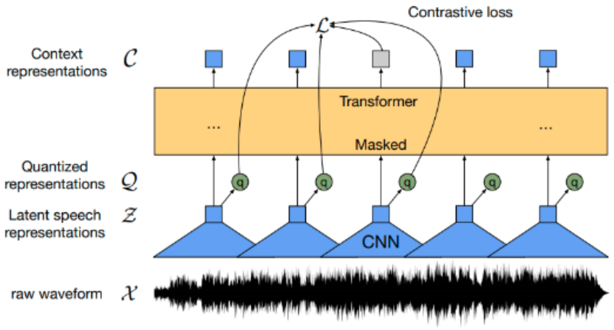
Advantages:

- **Self-supervised learning** so train without labelled data, low-resource: trained with 53k hours of unlabelled data, and just 10 minutes of labelled data. Trained on CommonVoice
- **Easy-implementation** with HuggingSound
- Used for **several languages** even without a lot of transcribed audios

Architecture:

CNN with 7 layers to learn the latent speech representations Z
Quantization: try to match the Z with codebooks, sort of phonemes
Transformers to learn contextual representations

Feature encoder: temporal convolution followed by a GELU activation function
Training: add noise to the audios so that the model learns to distinguish the voice
Decoder: 2 language models: 4-gram and transformer



Results

Announced: **WER=1.8** on clear data, **WER=3.3** on others

Our test: on 582 audios of 10s, **WER=21.7%** (because of the punctuation)

Conclusion

- Construct and test our own models
- Familiarize ourselves with traditional ASR
- Understand state-of-the-art algorithm

- Not usable results because insufficient processing power
- Transcriptions are very slow (Real Time Factor = 1)