Deep learning for Automatic Speech Recognition

Introduction

Automatic Speech Recognition (ASR), also known as speech to text, consists in converting spoken language as an audio signal into written text. While easy for humans, this task has historically been hard for machines.

Speech recognition is a very dynamic field of research because of its utility in our daily life. It can create **interaction between machines and humans**, no more in the language of the machine, but **in human languages**. The GAFAM have well understood the potential of the science, and work on algorithms for years, which are already applied to their products, such as Siri for Apple, Alexa for Amazon, the Google Home, ... In our project, we will try to implement our own deep-learning speech recognition algorithm and compare its results with the state-of-the-art algorithm.

Abstract

The objective of our work was to **understand** how **automatic speech recognition algorithms** work, to **implement** our own algorithms, and to **compare** them to the best current algorithm. We started our research by doing a state of the art on the subject. We first studied the **traditional ASR pipeline** and how **deep learning enabled significant progress,** with the example of **CTC.** We then implemented a **first CNN algorithm** and tested Meta's latest RSA algorithm: **Wav2Vec2.0**.

To carry out our learning, we used a **Kaggle dataset in French**, with a person reading French novels.
The metric chosen is the **Word Error Rate (WER)**, which allowed us to compare the performance of the different algorithms.

Kaggle Dataset

kaggle

French Single Speaker Speech Dataset [1]
8,600 audios of *Les Misérables* and *Arsène Lupin contre Herlock Sholmes* and their transcripts in French

Word Error Rate (WER)

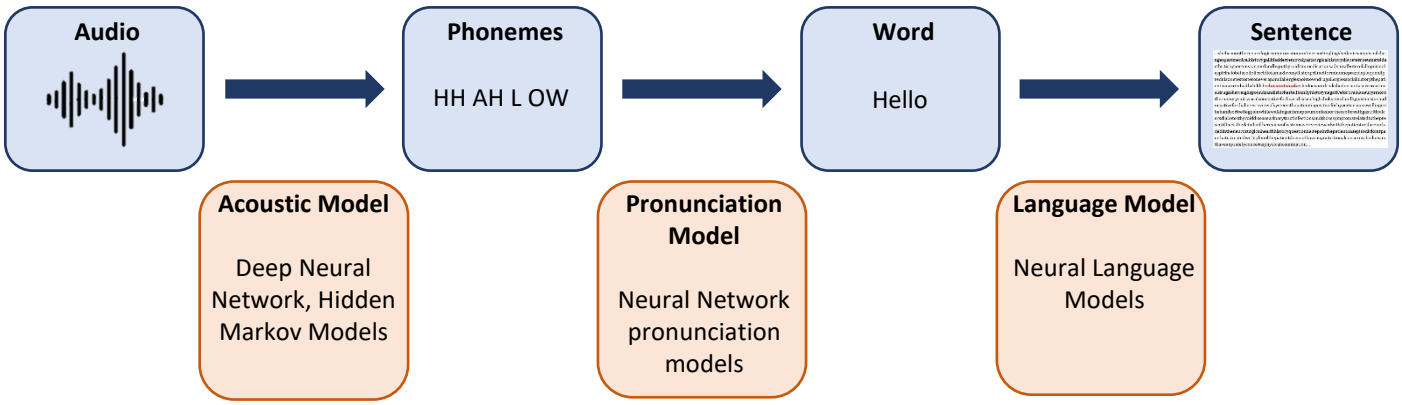
Very common metric to evaluate the performance of ASR, comparing a reference and a predict sentence using the formula:

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

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 [2]



Traditional ASR pipeline is composed of independent components, trained with different objectives
Error in one component may not behave well with error in another [3]
Can we train **end-to-end models** that include all these components?

First CNN model

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

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

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

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

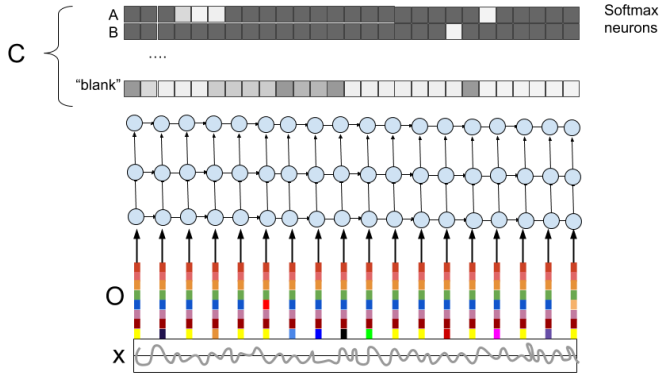
Results not usable because the learning time was too limited
WER > 11% in the literature

Deep learning for Automatic Speech Recognition

Towards end-to-end models: Connectionist Temporal Classification (CTC) [4]

Characteristics

- Trained on **labelled data** (i.e. transcribed audio)
- Usually 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

1. Pre-process raw audio to obtain a simple **spectrogram** (FFT of small windows of waveform) to get frequency content

2. Use **RNN**, whose output neurons encode a **distribution over symbols**

$$c \in \{A, B, \dots, space, blank\}$$

3. 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**

$$\boldsymbol{L(\theta) = \log \mathbb{P}\left(y^{*(i)} \middle| x^{(i)}\right) = CTC(c^{(i)}, y^{*(i)})}$$

Wav2vec2.0 [5]

- Developed by Meta
- 2019
- Authors: Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, Michael Auli

- Advantages:

  Self-supervised learning so train without labelled data, low-resource
  Easy-implementation with HuggingSound
  Used for **several languages** even without a lot of transcribed audios

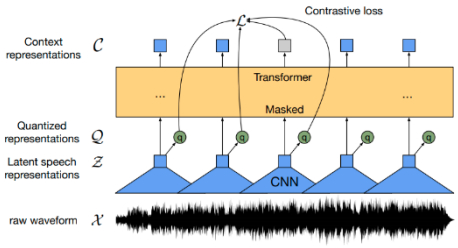
Architecture

1. CNN with 7 layers to learn the **latent speech representations Z**
2. **Quantization**: try to match the Z with codebooks, sort of phonemes
3. Transformers to learn **contextual rep.**

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

$$Loss = L_{contrastive} + L_{diversity} + L_{l_2}$$

Experiments: trained on CommonVoice, create 2 pre-trained models: BASE (dimension 768) and LARGE (dimension 1024)
Decoder: 2 language models: 4-gram and transformer



Results

With 10 mins of labelled data, 53k hours of unlabelled data: **WER = 5.7%** on clean audio, **10.1%** on noisy audio
22% better than previous algorithms
On Kaggle Dataset, with 582 audios of 10s, **WER = 21,7%**

Conclusion

In our study, we were able to familiarize ourselves with traditional ASR methods and seek to understand state-of-the-art algorithms. We were also able to develop a relatively simple **CNN algorithm**.

We tested this algorithm on the Kaggle dataset. Our **results are logically worse**, largely because it was **difficult for us to train** our models for a long time and on a lot of data.
The **Wav2Veq2.0 model was very impressive** in our tests, as it has very good results and is very easy to implement given . The variety of languages available is also a real advantage. However, the **transcriptions are slow** (10 seconds for a 10-second audio), which can be a hindrance to obtaining large audio transcriptions.

References

[1] Kaggle, Datasets, Bryanpark, French-Single-Speaker-Speech-Dataset
[2] Baevski, A., Hsu, W. N., Conneau, A., & Auli, M. (2021). Unsupervised speech recognition. Advances in Neural Information Processing Systems, 34, 27826-27839.
[3] Mohamed, Abdel-Rahman & Sainath, Tara & Dahl, George & Ramabhadran, Bhuvana & Hinton, Geoffrey & Picheny, Michael. (2011). Deep Belief Networks using discriminative features for phone recognition. Acoustics, Speech, and Signal Processing, 1988. ICASSP-88.
[4] Graves, Alex & Fernandez, Santiago & Gomez, Faustino & Schmidhuber, Jürgen. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. ICML 2006
[5] Schneider, S., Baevski, A., Collobert, R., & Auli, M. (2019). wav2vec: Unsupervised pre-training for speech recognition. arXivpreprintarXiv:1904.05862.