Artificial Intelligence

Automatic Speech Recognition

A cool application of AI!

Option 1: Introduction

Goals:
Gain a high-level understanding of ASR, with a focus on modern models like Whisper.
Explore Whisper in action.
Interested students can use linked resources to learn more.

What is ASR?

- Automatic Speech Recognition (ASR) is the use of machine learning to convert spoken language (audio signal) into written text
- Core AI application bridging audio ➔ natural language
- Powers tools/services like:
  - Voice assistants (Siri, Alexa)
  - Transcription tools (Otter.ai, Rev)
  - Accessibility services (closed captioning)

The Evolution of ASR

Brief History

1950s-70s	1980s-90s	2010s	2020s
Early pattern matching, Ex. IBM Shoebox Early systems could only recognize digits	**Acoustic Models: HMMs, GMMs** Made probabilistic modeling possible.	**Deep Learning revolution** Major breakthroughs with better sequence modeling	**Self-supervised + Large Models** Dominate ASR today. Ex. Wav2Vec, Whisper

Traditional ASR

Traditional approach to ASR involved 3 major parts:

1. Acoustic models like Hidden Mixture Models (HMMs) model sequences of sounds probabilistically. Gaussian Mixture Models (GMMs) estimate distributions over sound features.
2. Language models predict word sequences, originally with simple n-grams and now with deep learning.
3. feature engineering like Mel-frequency cepstral coefficients (MFCCs) manually extracted features from the audio.

Traditional to Modern ASR

- With **deep learning**, DNNs and later RNNs and transformers replaced many hand-engineered steps.
- Recently, **self-supervised learning**
  - Enables models to learn from raw audio without expensive labels

Modern ASR: End-to-End Systems

- Traditional ASR was very modular: separate models for Acoustic + Pronunciation + Language Models
- **End-to-End ASR**: Directly map audio to text with a single model, ex.
  - CTC loss: can deal with unaligned audio and text [CTC tutorial]
  - Attention-based encoder-decoders learn to align input and output dynamically [paper: Attention is All You Need]

Modern ASR: End-to-End Systems

- Advantages of end-to-end:
  - Less feature engineering
  - Simpler system design
  - Better with large data

Wav2Vec

- Self-supervised learning approach developed by Facebook AI in 2019 / 2020 (Wav2Vec2.0)
- Self-supervised learning on raw audio, not requiring expensive labeled datasets for pretraining
- Goal: Learn audio representations without needing transcripts

Wav2Vec

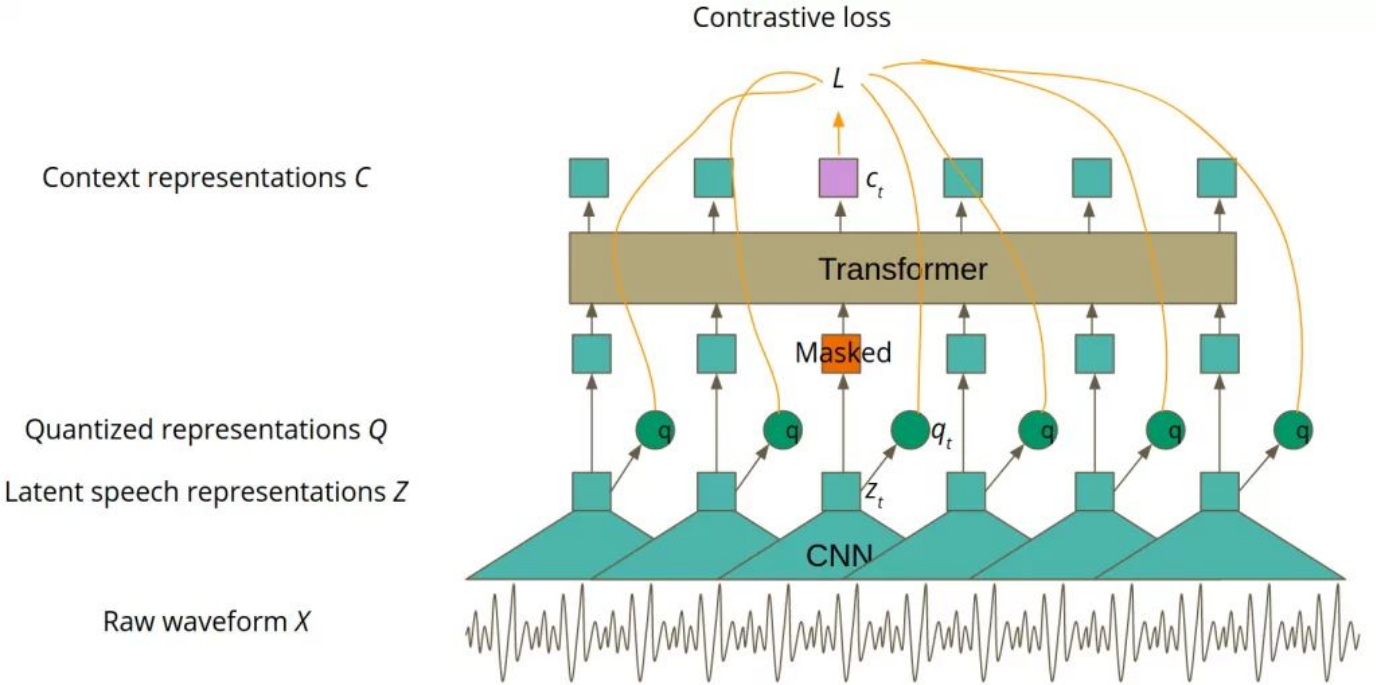
3 Main Phases of Wav2Vec2.0:

1. **Feature Encoder**: CNN encodes raw audio into lower-dimensional latent vectors
2. **Contextualizer**: Transformer network models long-range dependencies; understands longer chunks of speech
3. **Quantization & Contrastive Task**: Model must distinguish the true future latent vector from negative samples, forcing it to learn meaningful audio representations

After pretraining, fine-tune on a smaller labeled set.

Paper: "wav2vec 2.0" Blog: wav2vec

Wav2Vec 2.0 Architecture



- CNN reduces the raw input into dense features.
- Transformer captures sequence-level patterns like words or phrases.
- Quantization discretizes some representations, for the contrastive task.

Source: https://neurosys.com/blog/wav2vec-2-0-framework

Whisper

- Large-scale ASR Model developed by **OpenAI (2022)**
- Multilingual and multitask **supervised** training on 680k hours of audio
  - huge dataset! Much of it is noisy or machine- transcribed
- Capable of transcription, but also:
  - Translation
  - Language identification
  - Multilingual ASR

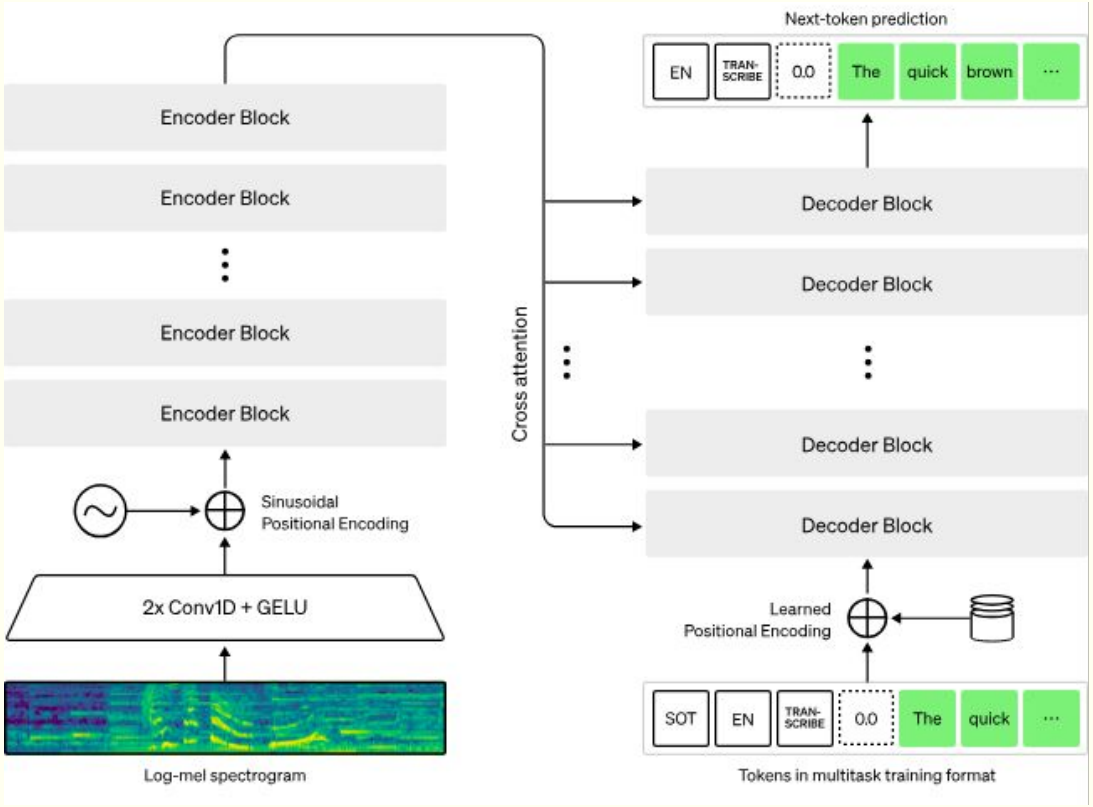
See the paper!

Whisper

- **Training set:**
  - Noisy, real-world diverse audio (YouTube, podcasts, etc.)
- **Label type:**
  - Human or machine-generated transcripts (ie. auto captions)
- This large-scale weak supervision ➔ model has robustness to real-world noise, accents, etc.
- **Result**: State-of-the-art performance without further fine-tuning

Whisper: Architecture

- End-to-end approach
- **Encoder-decoder Transformer**
- Audio split into chunks ➔ convert into log-Mel spectrogram ➔ encoder processes ➔ decoder generates tokens ➔ predicted text



Activity!

Whisper (large-v3 model) is hosted on HuggingFace through this web app: https://huggingface.co/spaces/openai/whisper

- Test out transcription by recording audio and transcribing it. How did it do? Are there any errors you noticed?

Activity!

Wav2vec and Whisper are available to download + experiment with via Hugging Face [repository]

- Also available through OpenAI endpoints (not free)

Activity: Use Whisper via Python! Open the associated Jupyter notebook with this lecture (in the Github repo) and complete the activities.

Applications of ASR

- Voice search
- Dictation software
- Automatic captioning for meetings and videos
- Healthcare: clinical note-taking with an automatic scribe
- Customer service: call centers

Challenges in ASR

- Accents, dialects can be difficult to accurately transcribe
- Background noise can be conflated with speech
- Code-switching (mixing languages)
- Domain adaptation to domains such as medical and legal, where jargon is common

Activity!

Go back to the hosted Whisper model.
Pick your favorite domain (ex. medical, legal, technical) and try to “break” the model.
What words did Whisper struggle with transcribing accurately?
Can you find another way to “break” the model?