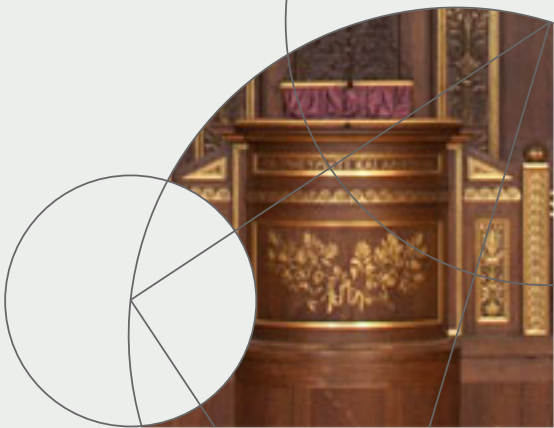


# Data Acquisition and Processing II: Deep Learning @ Speech Recognition

Rensheng Wang,
rwang1@stevens.edu
Dept. of Electrical and Computer Engineering
Stevens Institute of Technology

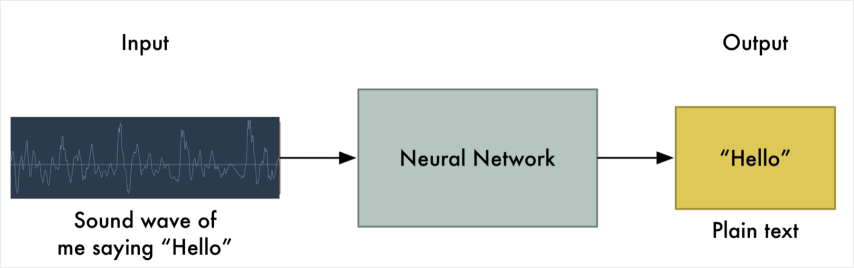


# Speech Recognition in Deep Learning

- Speech recognition is invading our lives. It's built into our phones, our game consoles and our smart watches.
- It's even automating our homes. For just $50, you can get an Amazon Echo Dot  a magic box that allows you to order pizza, get a weather report or even buy trash bags  just by speaking out loud.
- But speech recognition has been around for decades, so why is it just now hitting the mainstream?
- The reason is that deep learning finally made speech recognition accurate enough to be useful outside of carefully controlled environments.

# Deep Learning Is Not Always a Black Box

- If you know how neural machine translation works, you might guess that we could simply feed sound recordings into a neural network and train it to produce text:



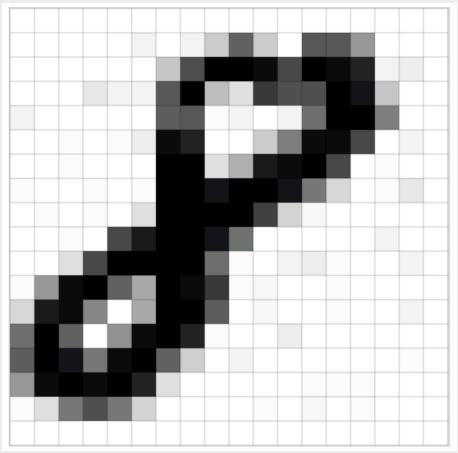

- That's the holy grail of speech recognition with deep learning, but we aren't quite there yet.

  The big problem is that speech varies in speed. One person might say "hello!" very quickly and another person might say "heeeellllllllllllooooo!" very slowly, producing a much longer sound file with much more data.

- Both both sound files should be recognized as exactly the same text - "hello!" Automatically aligning audio files of various lengths to a fixed-length piece of text turns out pretty hard.

# Turning Sounds into Bits

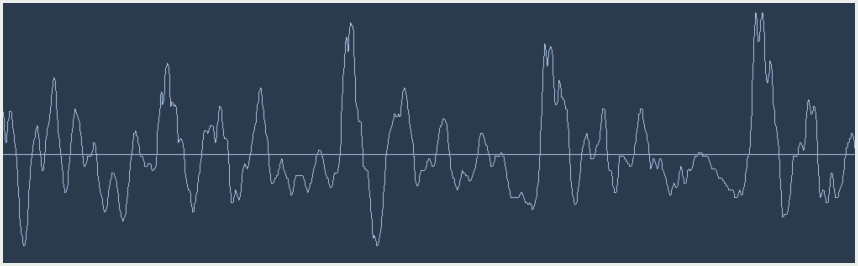
- The first step in speech recognition is obvious we need to feed sound waves into a computer.
- In previous lectures, we learned how to take an image and treat it as an array of numbers so that we can feed directly into a neural network for image recognition.



| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 12 | 0 | 11 | 39 | 137 | 37 | 0 | 152 | 147 | 84 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 41 | 160 | 250 | 255 | 235 | 162 | 255 | 238 | 206 | 11 | 13 | 0 |
| 0 | 0 | 0 | 16 | 9 | 9 | 150 | 251 | 45 | 21 | 184 | 159 | 154 | 255 | 233 | 40 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 145 | 146 | 3 | 10 | 0 | 11 | 124 | 253 | 255 | 107 | 0 | 0 |
| 0 | 0 | 3 | 0 | 4 | 15 | 236 | 216 | 0 | 0 | 38 | 109 | 247 | 240 | 169 | 0 | 11 | 0 |
| 1 | 0 | 2 | 0 | 0 | 0 | 253 | 253 | 23 | 62 | 224 | 241 | 255 | 164 | 0 | 5 | 0 | 0 |
| 6 | 0 | 0 | 4 | 0 | 3 | 252 | 250 | 228 | 255 | 255 | 234 | 112 | 28 | 0 | 2 | 17 | 0 |
| 0 | 2 | 1 | 4 | 0 | 21 | 255 | 253 | 251 | 255 | 172 | 31 | 8 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 4 | 0 | 163 | 225 | 251 | 255 | 229 | 120 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 |
| 0 | 0 | 21 | 162 | 255 | 255 | 254 | 255 | 126 | 6 | 0 | 10 | 14 | 6 | 0 | 0 | 9 | 0 |
| 3 | 79 | 242 | 255 | 141 | 66 | 255 | 245 | 189 | 7 | 8 | 0 | 0 | 5 | 0 | 0 | 0 | 0 |
| 26 | 221 | 237 | 98 | 0 | 67 | 251 | 255 | 144 | 0 | 8 | 0 | 0 | 7 | 0 | 0 | 11 | 0 |
| 125 | 255 | 141 | 0 | 87 | 244 | 255 | 208 | 3 | 0 | 0 | 13 | 0 | 1 | 0 | 1 | 0 | 0 |
| 145 | 248 | 228 | 116 | 235 | 255 | 141 | 34 | 0 | 11 | 0 | 1 | 0 | 0 | 0 | 1 | 3 | 0 |
| 85 | 237 | 253 | 246 | 255 | 210 | 21 | 1 | 0 | 1 | 0 | 0 | 6 | 2 | 4 | 0 | 0 | 0 |
| 6 | 23 | 112 | 157 | 114 | 32 | 0 | 0 | 0 | 0 | 2 | 0 | 8 | 0 | 7 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Turning Sounds into Bits

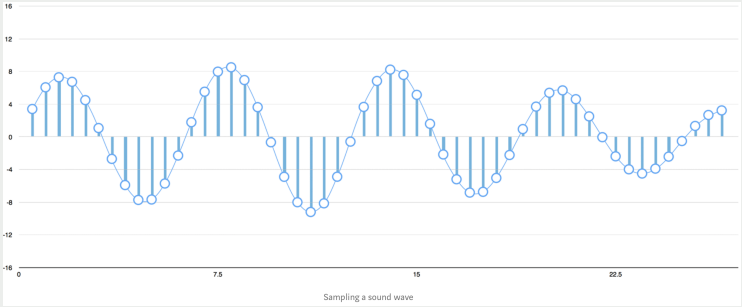
- But sound is transmitted as waves. How do we turn sound waves into numbers? Lets use this sound clip of me saying Hello:



- Sound waves are one-dimensional. At every moment in time, they have a single value based on the height of the wave.

# Sampling

- To turn this sound wave into numbers, we just record of the height of the wave at equally-spaced points:



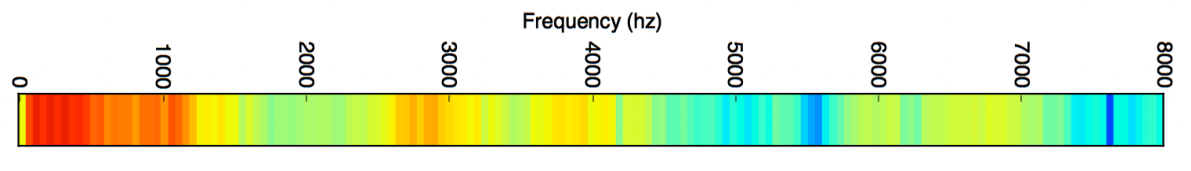

- This is called sampling. We are taking a reading thousands of times a second and recording a number representing the height of the sound wave at that point in time.
- "CD Quality" audio is sampled at 44.1khz (44,100 readings per second). But for speech recognition, a sampling rate of 16khz (16,000 samples per second) is enough to cover the frequency range of human speech.

# Sampling of Sound Wave

- Lets sample our Hello sound wave 16,000 times per second. Heres the first 100 samples:

```
[-1274, -1252, -1160, -986, -792, -692, -614, -429, -286, -134, -57, -41, -169, -456, -450, -541, -761, -1067, -1231, -1047, -952, -645, -489, -448
, -397, -212, 193, 114, -17, -110, 128, 261, 198, 390, 461, 772, 948, 1451, 1974, 2624, 3793, 4968, 5939, 6057, 6581, 7302, 7640, 7223, 6119, 5461,
 4820, 4353, 3611, 2740, 2004, 1349, 1178, 1085, 901, 301, -262, -499, -488, -707, -1406, -1997, -2377, -2494, -2605, -2675, -2627, -2500, -2148, -
1648, -970, -364, 13, 260, 494, 788, 1011, 938, 717, 507, 323, 324, 325, 350, 103, -113, 64, 176, 93, -249, -461, -606, -909, -1159, -1307, -1544]
```

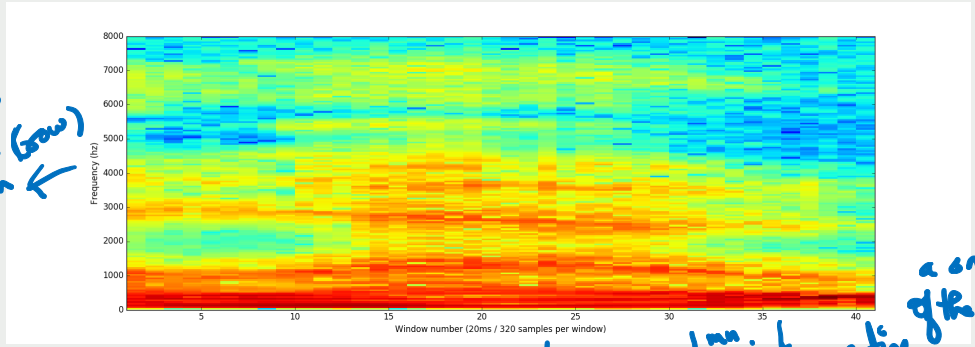
- We use Fourier transform to convert the time-domain samples to a Frequency Domain vector. For example, take the first 20 millisecond sound samples and take the FFT, you have




where each Frequency sample is shown as a color, the warm color "red" indicates larger values while the cold color "blue" as smaller values.

# Convert Sound Samples to Spectrogram

- If we repeat this process on every 20 millisecond chunk of audio, we end up with a spectrogram (each column from left-to-right is one 20ms chunk):




this comes from FFT (row) if signal done on ←

↳ every col mn is truncating a small part of the time series

- In this spectrogram, $y$-axis is the frequency domain vector shown as colors while $x$-axis is the time. Each column or each bin in the plot is the Fourier transform vector magnitude of a 20 millisecond time length audio signal samples. Over time line, we cut every 20 millisecond audio signal and convert to a vertical color bar. Padding all the 20 millisecond signal frequency domain magnitude vectors over time, we obtain the above spectrogram.

# Recognizing Characters from Short Sounds

- Now that we have our audio in a format thats easy to process, we will feed it into a deep neural network.
- The input to the neural network will be 20 millisecond audio chunks. For each little audio slice, it will try to figure out the letter that corresponds the sound currently being spoken.