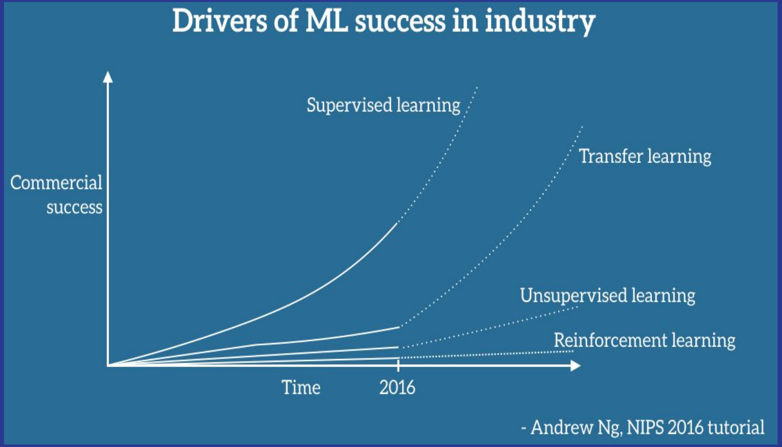
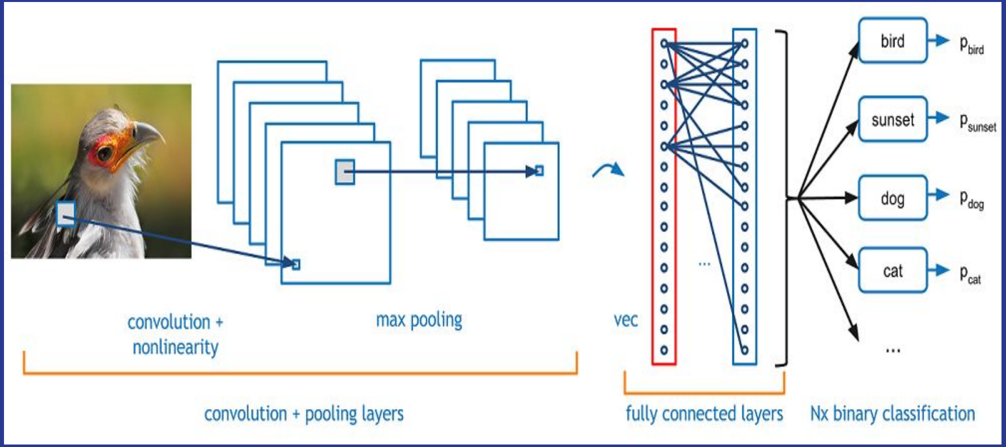
Speech Recognition

A. Abdelmoneim, G. Guz, I. Del Rio, D. Wu, Y. Zhuk
a.k.a.
Dambridge 🅱nalytica

Overview

- Nowadays, 90% of ML applications in the industry involve supervised learning, which is attempting to solve some sort of a classification/forecasting problem
    - Given an image, detect which animal is depicted
    - Given time series of stock prices, predict the price at the next timestep

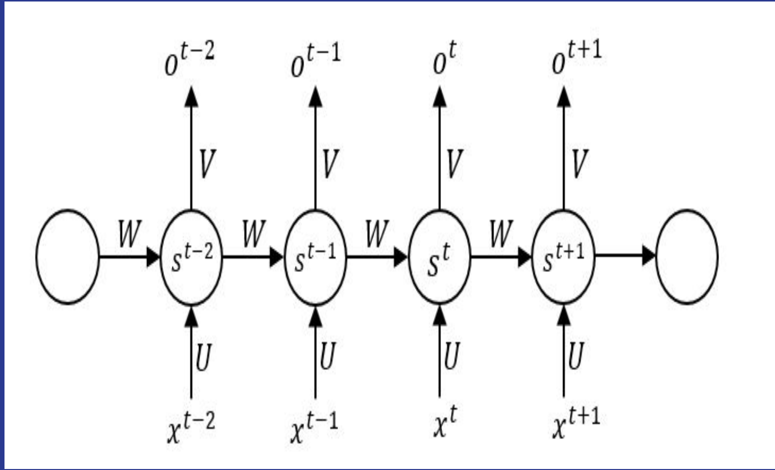




Sequential data format

- To work with sequential data, we use Recurrent Neural Networks. They allow us to "look in the past":
  - At timestep t, we feed in a data vector x^t and multiply it by matrix U. This is called "embedding": the network represents data in the format it will use in future computations.
  - Using some matrix operations and nonlinear functions, update vector s^t, which represents the state
  - Using s^t and V, produce the output o^t

- Hence our parameters are:
  - U, V are learned, fixed when predict
  - W is updated at each timestep



Temporal classification

- A lot of problems in Natural Language Processing involve problems where input and output are of different size:
  - For translation, the input sentence and its translation can have different sizes

- In speech recognition, the signal is usually split into small chunks (~15ms each). Thus, the labellings for each of these chunks are generated, and we need to find the way to combine them.

Some theory: language models

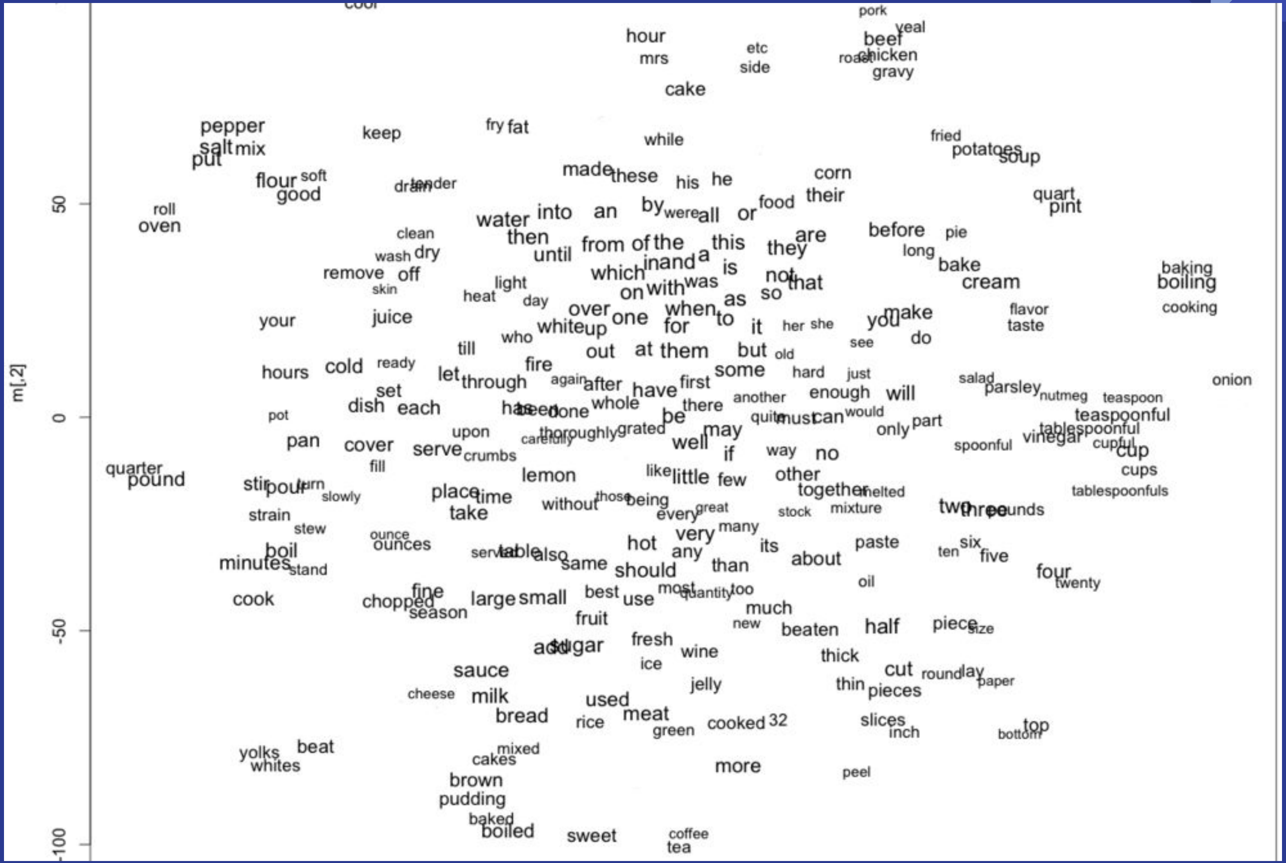
- Most NLP systems require an ability to represent words as vectors, in order to do math on them

- We need to construct these vectors so that they give us information about the structure of a language

- Given a context (first N words of a sentence), what is the most probable next word?

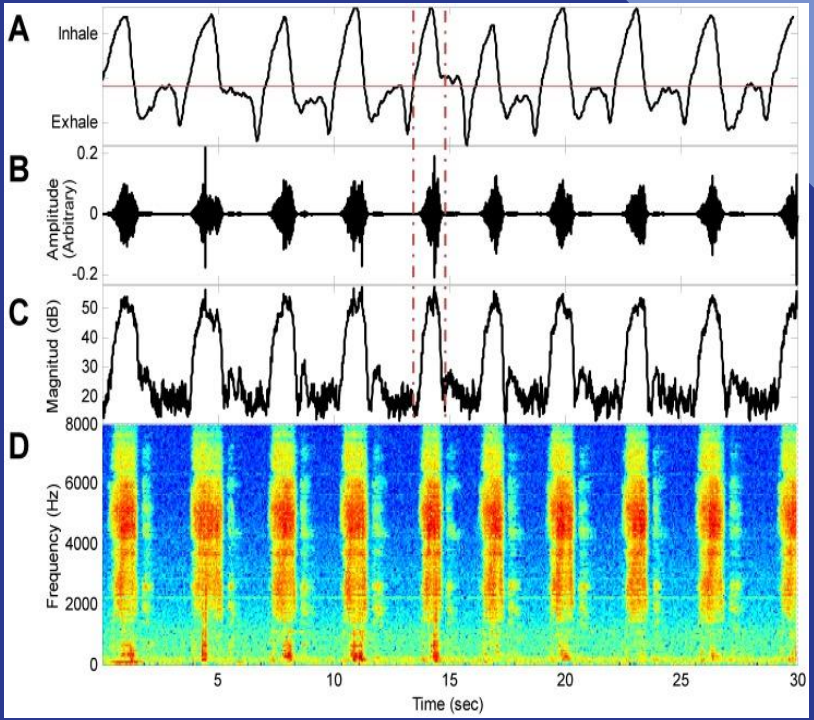
- Common examples
    - N-gram (co-occurrence counts)
    - Word2vec
    - GloVe

Some theory: language models



Data

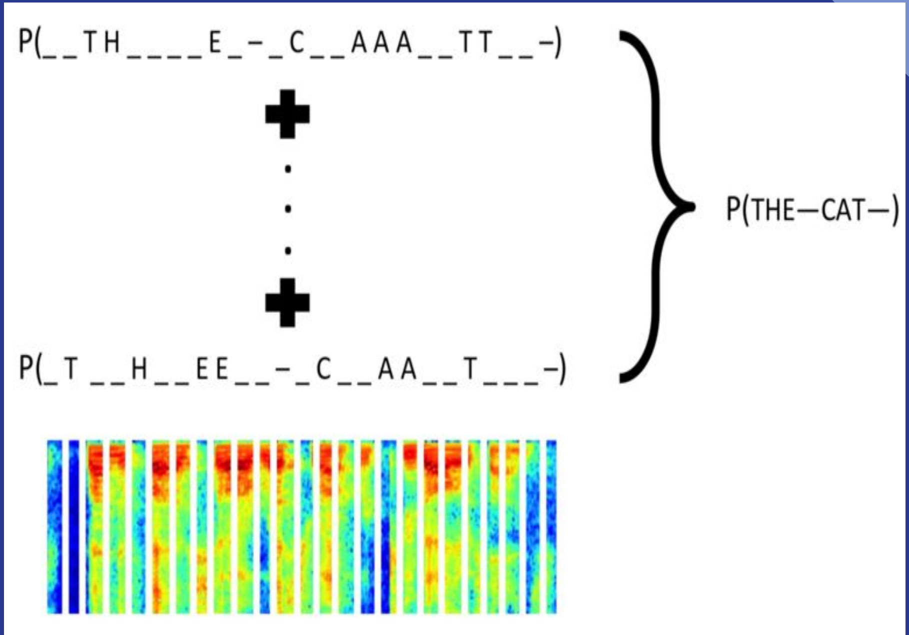
- For speech recognition, the audio signal is usually converted into features using Fast Fourier Transform algorithm, which decomposes the signal (which is just Amplitude vs. Time function) into a linear combination of sinusoids.
- Output is a sequence consists of n-dimentional vectors. Each element of the vector specifies “how much” of the sinusoid with i-th frequency there is in the signal



Approaches

Connectionist Temporal Classification (CTC)

- Check out https://distill.pub/2017/ctc/
- We are given an alphabet of English letters with a blank symbol
  A = { a, b, …, z, _ }
- Encode: RNN outputs the distributions for each FFT element, i.e. the probability for each letter of “being said” at this signal segment
- Decode: Using a dynamic programming algorithm, scan the probabilities and output the most probable labelling (hint: it is not just taking the highest-probability symbol at each timestep)



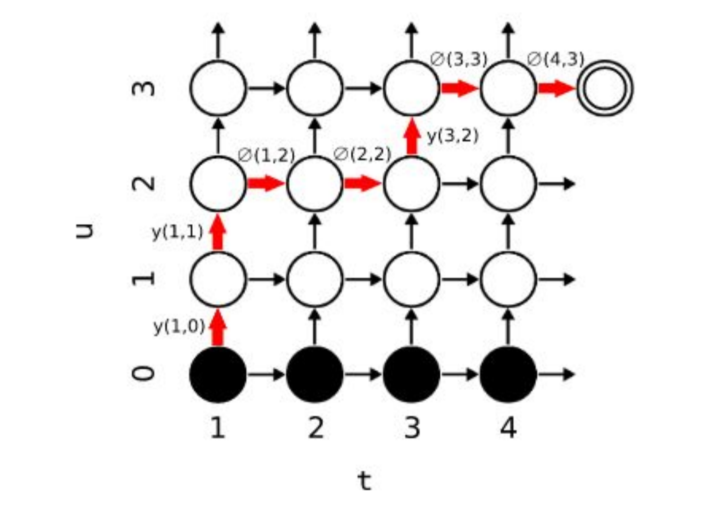
CTC: Results

Prediction:	Witthoit the dataset the article asele sa
Ground truth:	Without the dataset the article is useless

Prediction:	Be careful with your prognostigations saia
Ground truth:	Be careful with your prognostigations said the stranger

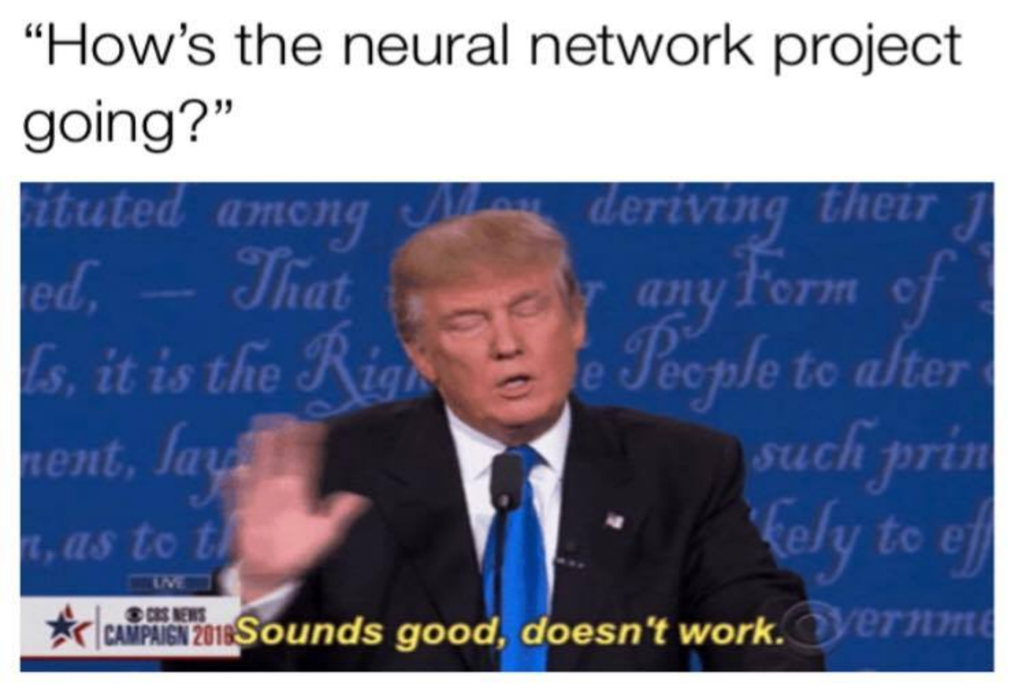
RNN Transducer

- Runs RNN on sequence of FFTs (like CTC).
- Runs RNN on the predictions so far. For example, if the most probable sequence so far is “hi launch”, we will run the second RNN on it
- Very flexible, hacky and hard to train



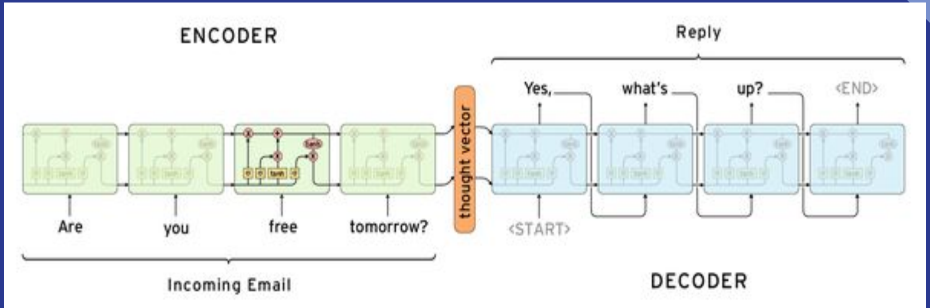
Transducer: results

Didn’t work :-(



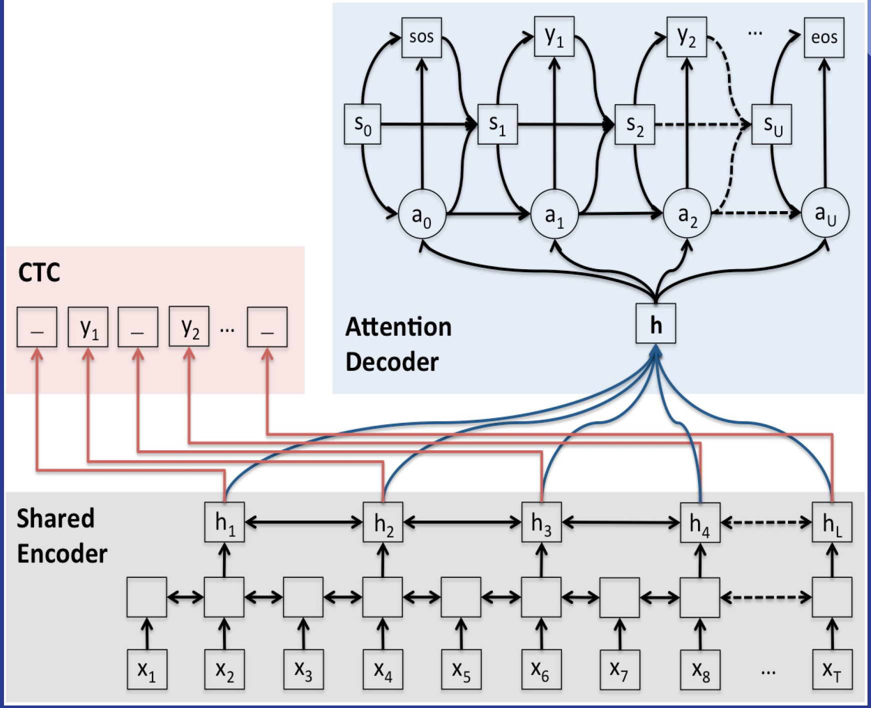
Encoder Decoder Networks

- An encoder is a network that takes your input and outputs a special representation (called a feature vector) of your input.
- The decoder is again a network (usually the same network structure as encoder but in opposite orientation) that takes the feature vector from the encoder and produces an output.



Sequence-to-Sequence models with Attention

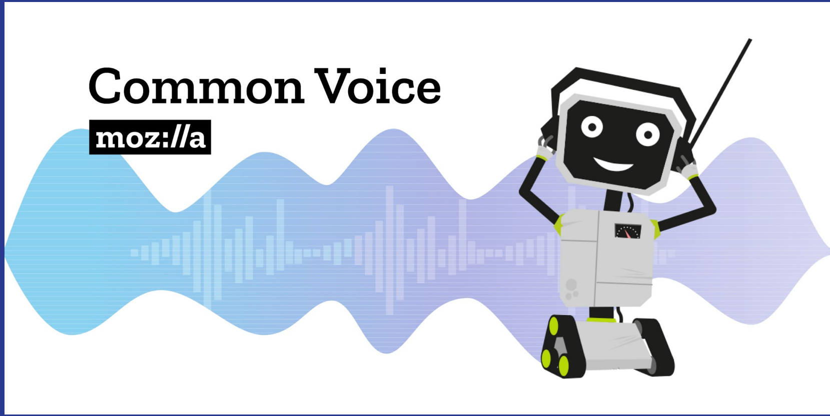
- Encoder is an RNN that generates embeddings which take context into account
- The decoder uses an attention mechanism to select embeddings from different timesteps and generate words from them
- It selects the previous timesteps based by taking a weighted linear combination of them, where the weights are learned



Dataset

Mozilla Common Voice Dataset

- An open source dataset with approximately 500 hours of speech
- Consists of data from both genders, multiple age ranges, and multiple accents



Storage

- The dataset is BIG (training data is approximately 85GB)
- We’re storing the data using a Hierarchical Data Format (HDF5)
- Allows for very efficient reading and writing of data
- Allows for sliced reading of data from disk, so particular subsets of the data can be extracted for processing, which is critical due to limited RAM
- Random reads get *very* slow as the file’s size increases

The End