Speech

Arthur J. Redfern
arthur.redfern@utdallas.edu

Apr 08, 2019
Apr 10, 2019

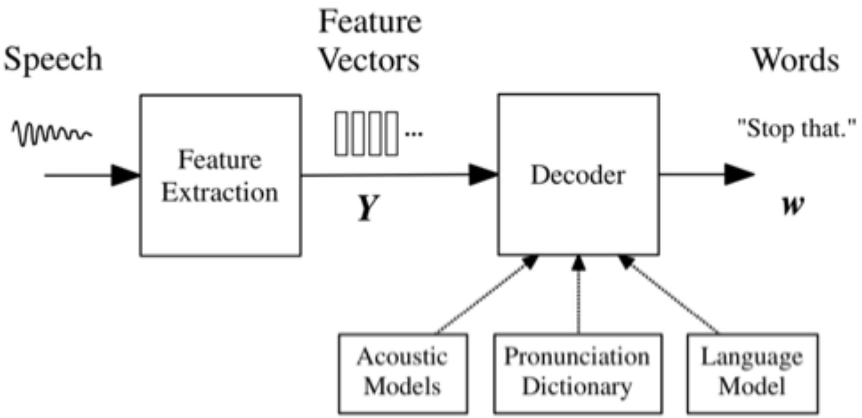
Outline

- Motivation
- Speech and audio
- Pre processing
- Network structures
- Speaker identification
- Keyword spotting
- Conditional modeling
- Speech to text
- Text to speech
- References

Motivation

Classical Speech To Text

- Speech or audio waveform
  - $x(n)$, $n = 0, \ldots, N - 1$
- Feature vectors
  - $\mathbf{Y} = [\mathbf{y}_0 \ldots \mathbf{y}_{T-1}]$
  - MFCC, Δ, ΔΔ with ~ 40 total features are common
- Word sequence **w**
  - $\mathbf{w}_{hat} = \arg\max_{\mathbf{w}} (P(\mathbf{w} \mid \mathbf{Y})) = \arg\max_{\mathbf{w}} (P(\mathbf{Y} \mid \mathbf{w}) P(\mathbf{w}))$
  - The acoustic model determines $P(\mathbf{Y} \mid \mathbf{w})$ and uses ~ 44 phonemes for the English language (24 cons, 20 vowel)
  - Phonemes are concatenated via a pronunciation dictionary to make words **w**
  - The language model determines $P(\mathbf{w})$; N gram language models that estimate the probability of a word given the previous N – 1 words are commonly used



For a nice overview see
- Speech recognition
  https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%209%20-%20Speech%20Recognition.pdf
- Deep audio
  http://slazebni.cs.illinois.edu/spring17/lec26_audio.pdf

Figure from https://mi.eng.cam.ac.uk/~mjfg/mjfg_NOW.pdf

Classification Problems

- Motivation
- Speech and audio
- Pre processing
- Network structures
- Speaker identification
- Keyword spotting
- Speech to text
- Text to speech
- References

Common speech problems are classification problems at their core: map a sound waveform to a finite number of classes

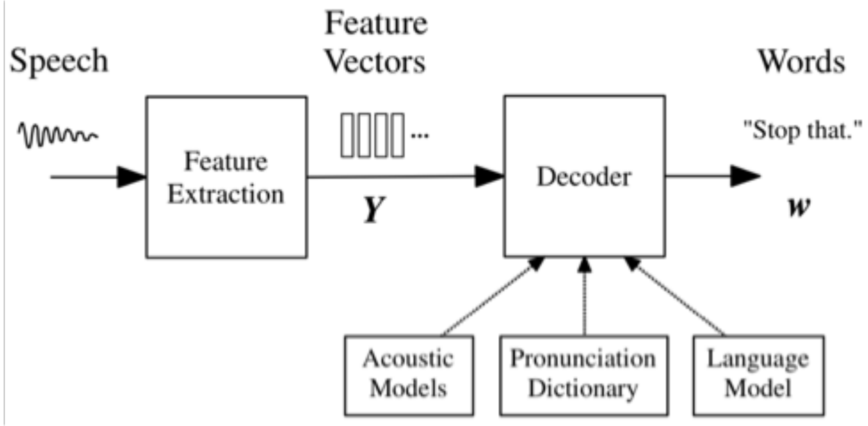


Figure from https://mi.eng.cam.ac.uk/~mjfg/mjfg_NOW.pdf

Generation Problems

- Motivation
- Speech and audio
- Pre processing
- Network structures
- Speaker identification
- Keyword spotting
- Speech to text
- Text to speech
- References

Speech also includes common generation problems

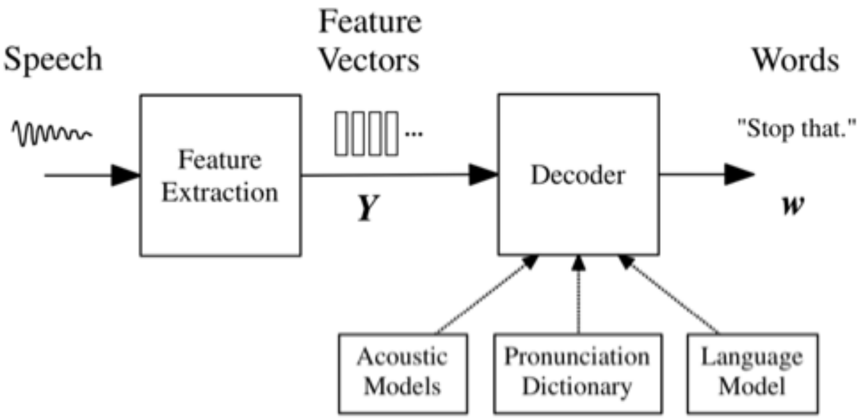


Figure from https://mi.eng.cam.ac.uk/~mjfg/mjfg_NOW.pdf

The Strategy Described Here

- Use xNN based methods for feature extraction and prediction
  - xNNs are universal approximators
  - Most speech classification problems can be cast as approximating a mapping from sound waveforms to classes
  - Sometimes use RNNs to combine information in a sequential nature
  - Sometimes use CNNs to combine information across a window of time and frequency
  - Train end to end

Disclaimer

- There’s a lot of speech related stuff not included here
  - Different methods within the categories of problems included here
  - Problems that are not included here
- Possibly some of this will be addressed in future versions of the slides
- Regardless of whether it is or not, hopefully these slides provide enough of a base from which to branch off and learn more on your own

Speech And Audio

The Human Speech And Audio Chain

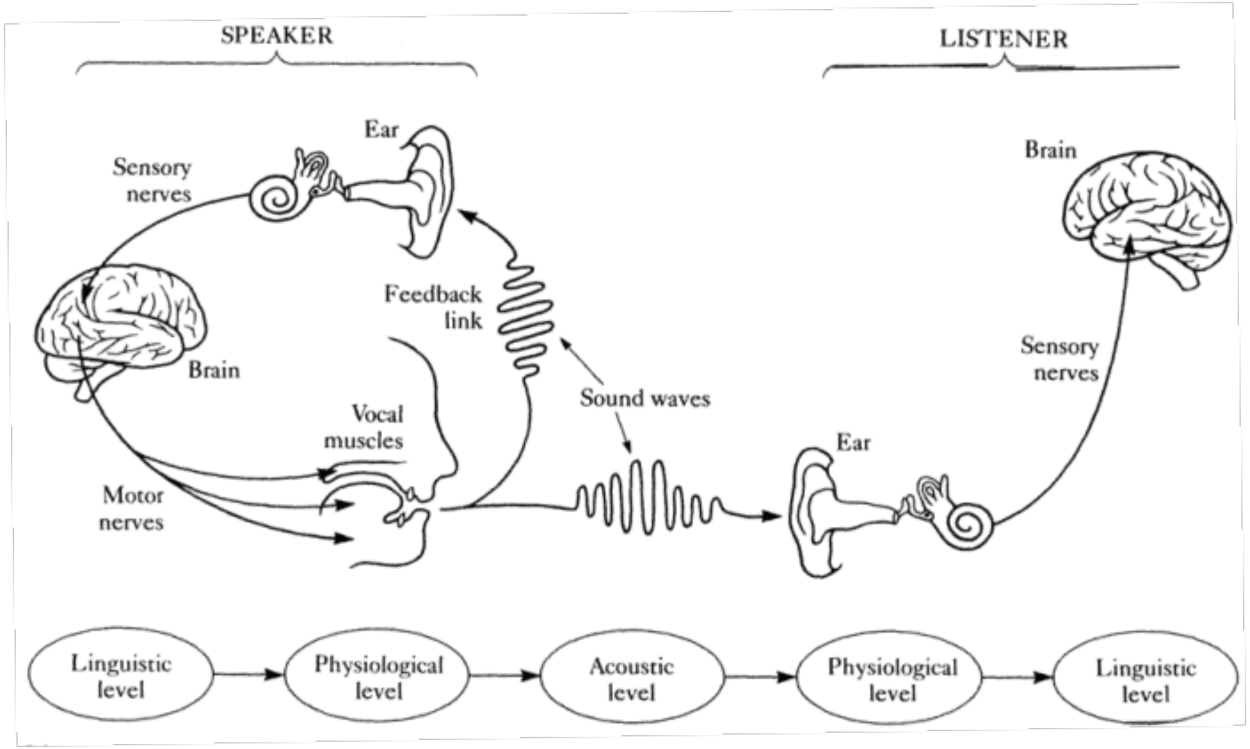
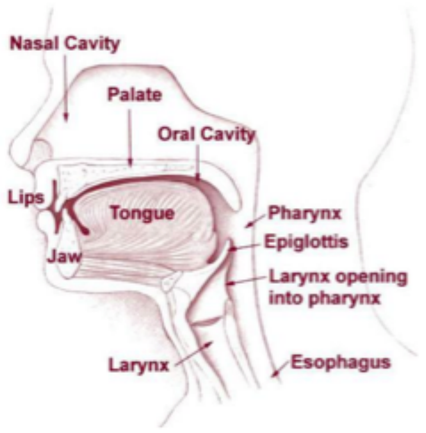
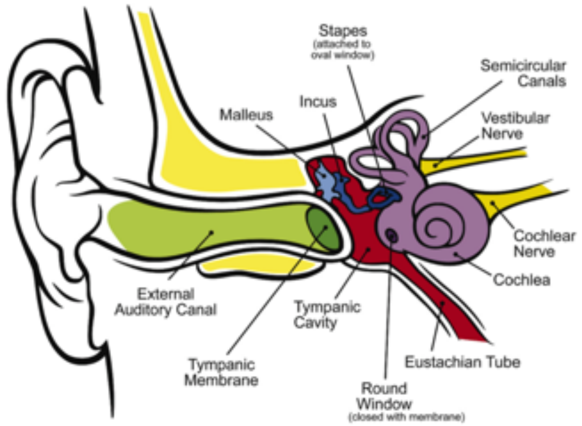
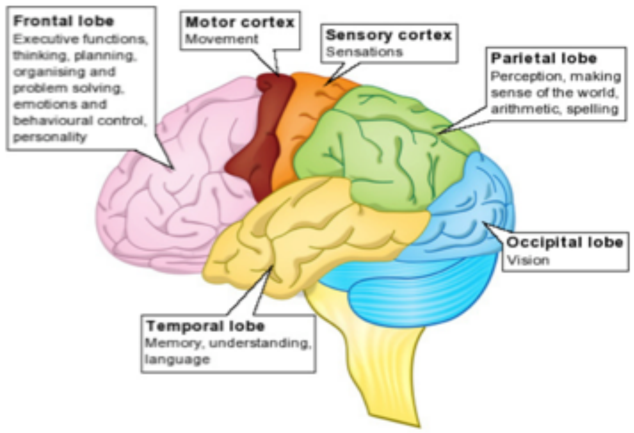


Figure from http://www.columbia.edu/~rmk7/HC/HC_Readings/Denes_Pinson1-2.PDF

Generation, Perception And Understanding







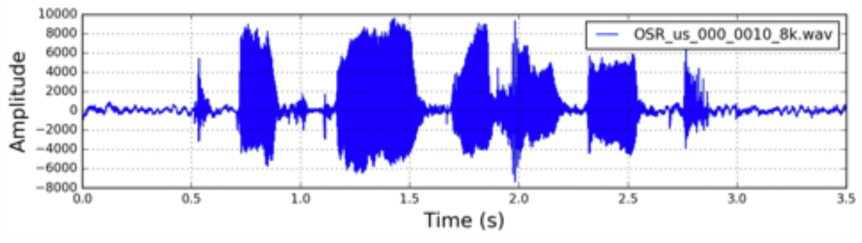
The Machine Speech And Audio Chain

- Synthesis
  - Information
  - Text to digital speech waveform
  - DAC and amplifier
  - Speaker
- Sound waveform
- Analysis
  - Microphone
  - Amplifier and ADC
  - Digital speech waveform to text
  - Information

Pre Processing

Time Domain Waveform

- The microphone is a transducer that converts sound waves to continuous time voltages
- Continuous time voltages are sampled by an ADC to create discrete time domain samples
  - Typically the ADC produces a specific number of bits per sample at a particular rate
  - A continuous real baseband signal bandlimited to B Hz can be reproduced exactly via samples at 2B Hz
  - This would imply no loss of information
- Humans can hear sounds from ~ 20 – 20 kHz so sampling frequencies would need to be > 40 kHz to prevent a loss of information for humans
  - For reference CDs are sampled at 44.1 kHz at 16 bits
  - Speech datasets for machine learning are frequently sampled at 4, 8 or 16 kHz

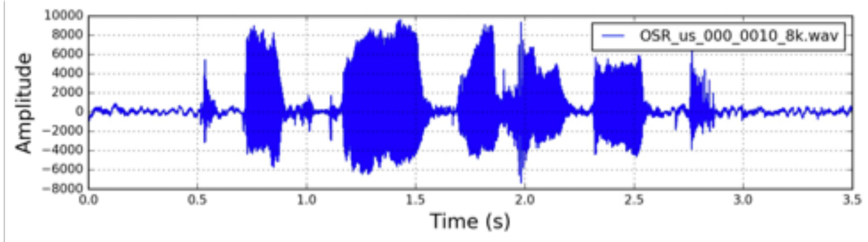


Signal in the Time Domain

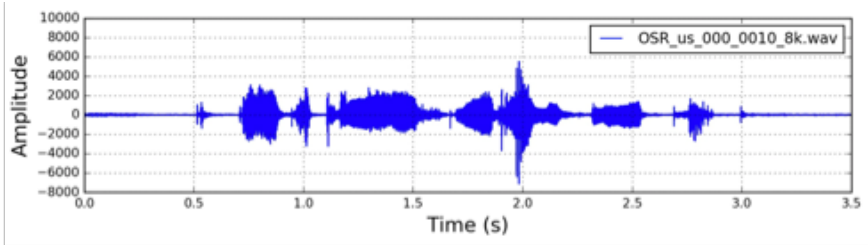
Figure from https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

Pre Emphasis

- To help with later processing a pre emphasis filter is sometimes applied at this point to increase high frequency components
    - High frequency components tend to be smaller than low frequency components for human created sounds
- Not all systems include pre emphasis



Signal in the Time Domain

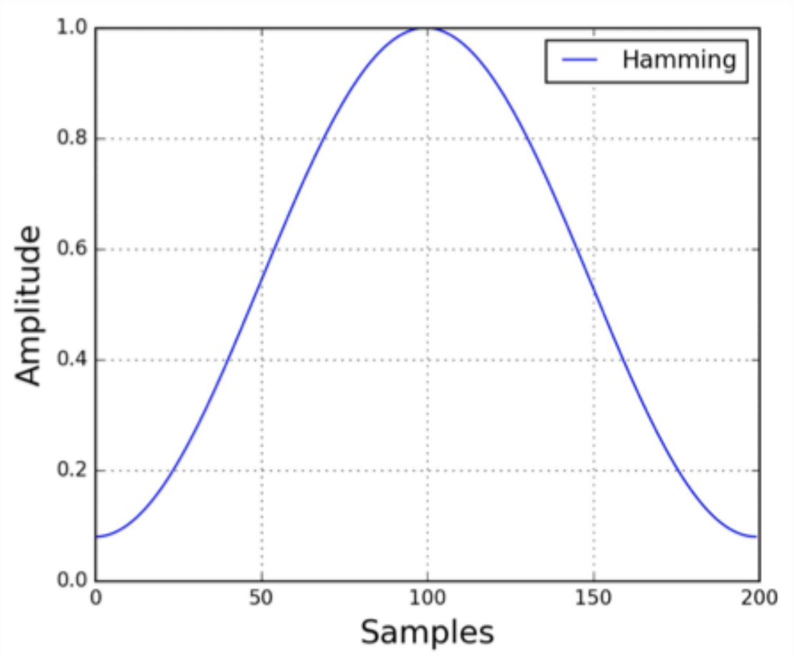


Signal in the Time Domain after Pre-Emphasis

Figure from https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

Windowing And Spectrogram

- To better understand the speech or audio signal it's useful to look at it in the frequency domain
  - But taking a DFT of the whole waveform would lose all temporal information
- So the input waveform is blocked into frames of ~ 20 – 25 ms with ~ 10 ms of overlap
  - Blocking is equivalent to windowing by a rectangular function which has a sinc() for a transform and results in spectral leakage (multiplication in the time domain == convolution in the frequency domain)
  - To reduce the spectral leakage a Hamming window is typically applied (lower side lobes in the freq domain)
- The DFT of each frame is taken and the magnitude is used to create the spectrogram



Hamming Window

Figure from https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

MFCC

Mel frequency cepstral coefficients; note that there are variations of this

- Humans don't respond to all frequencies equally
  - The mel frequency spacing mimics the greater sensitivity of the ear to lower frequencies
  - A mel filter bank with ~ 40 filters can be applied to the spectrogram output to create a mel filter bank output
  - For log mel take the log of the mel spectrogram

- Next steps
  - Take the DCT of this to concentrate energy
  - Keep coefficients 2 – 13 (replace coefficient 1 with the log energy, throw away coefficients 14+ as most information is in lower frequencies)
  - Create additional features with Δ values representing 1st order differences in coefficients 1 – 13 (~ 1st order derivative, provides shape info)
  - Create additional features with ΔΔ values representing 1st order differences in Δ values (~ 2nd order derivative, provides shape info)
  - Normalize coefficients to 0 mean and 1 variance

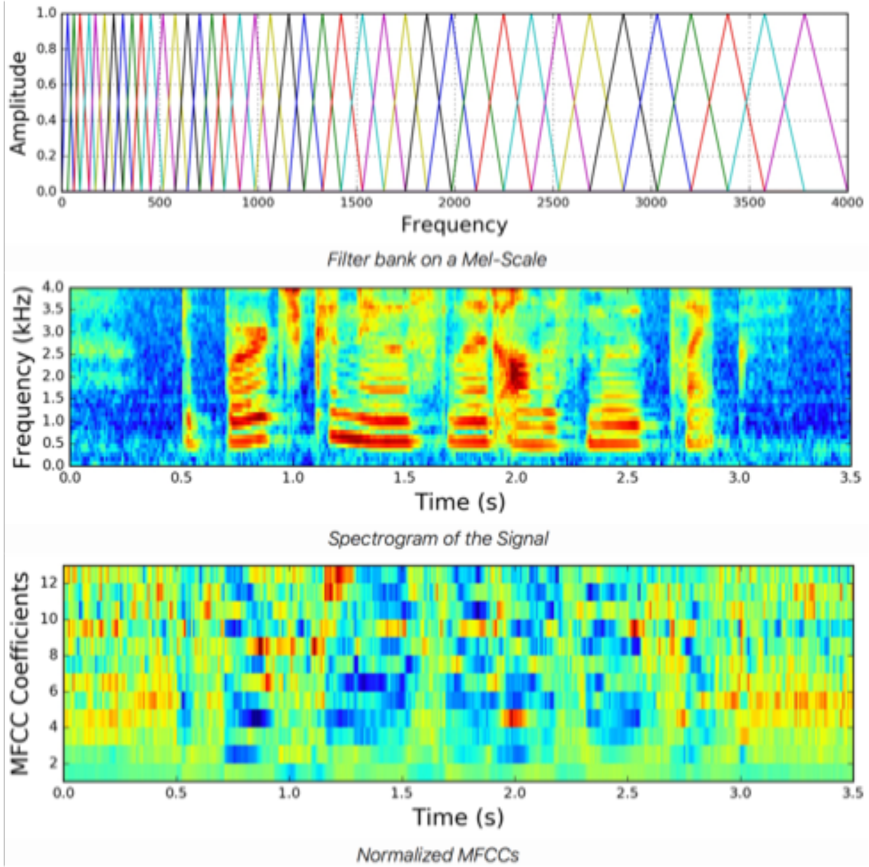
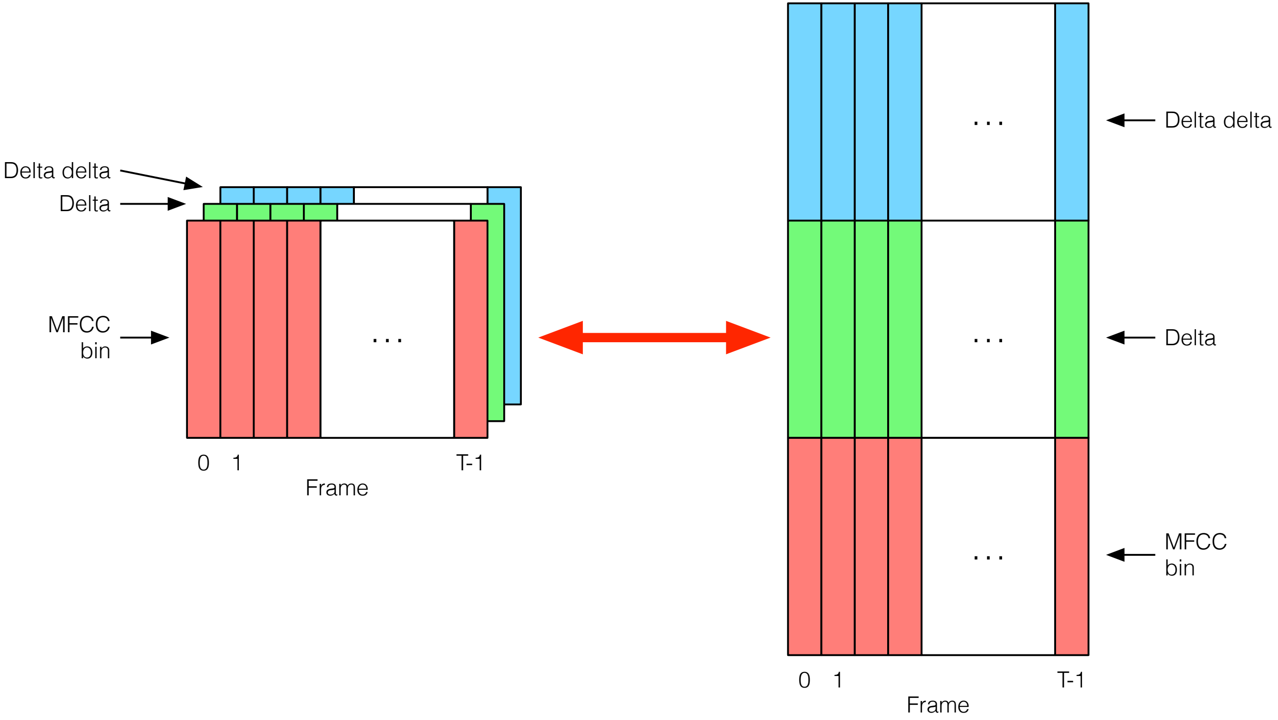


Figure from https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

Network Structures

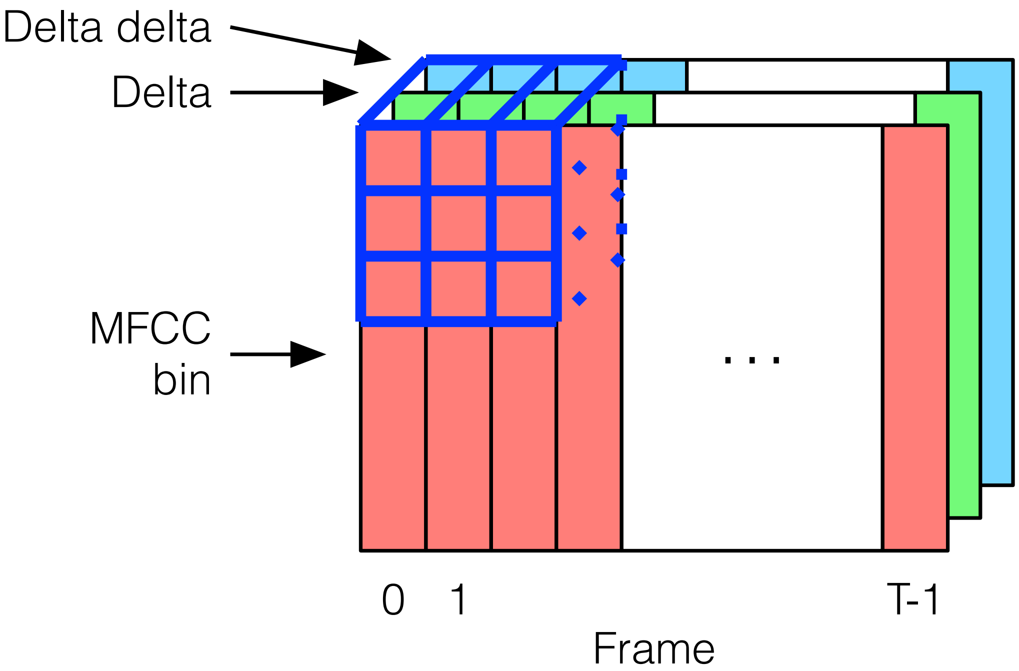
Different Views Of Pre Processed Speech

A 2D image of derivatives x bins x frames or a sequence of 1D vectors of stacked derivatives and bins



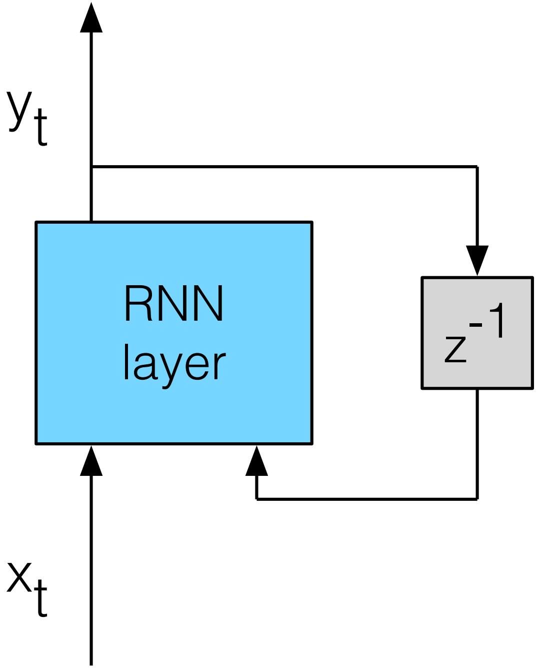
CNNs

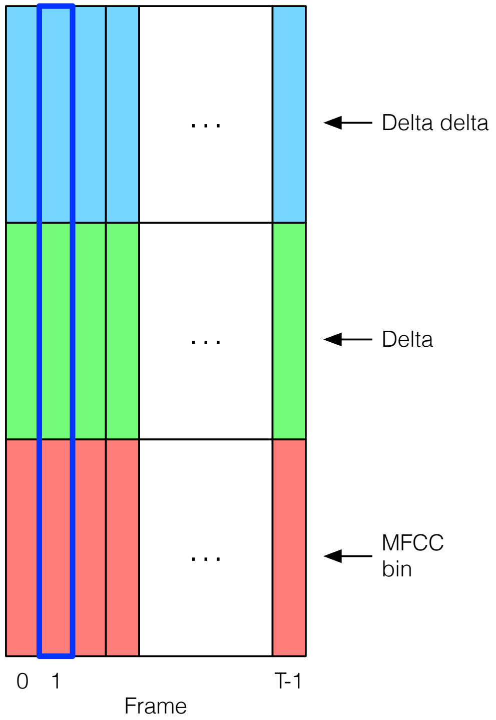
CNNs exploit translation invariance in space (in this case time and bin) for efficiency to combine info across time and bin to create stronger features



RNNs

RNNs exploit sequential structure in time (frame) to map from weaker features to stronger features





Should You Use A CNN Or RNN / Variant?

- For the problem of interest
  - Which is better for mapping from weak features to strong features with respect to accuracy?
  - Which is better for mapping from weak features to strong features with respect to efficiency of implementation?
  - What matters?

- The traditional split has been CNNs for vision and RNN / variants for speech and language
  - But there are blurring lines and CNNs are finding more uses in problems with sequential data
  - An argument can be made that RNNs / LSTMs are not the most efficient structure for propagating longer range information and are also inefficient to implement as they're memory bound
  - Attention based models are also starting to see more applications in speech
  - For more commentary see: The fall of RNN / LSTM (https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0)
  - Also see: When recurrent models don't need to be recurrent (https://bair.berkeley.edu/blog/2018/08/06/recurrent/)

- It will be interesting to see how this evolves

Speaker Identification

Recognition And Verification

- Recognition
  - Identify which of (potentially) many speakers is speaking
- Verification
  - Identify if a person is who they say they are or not
- Data
  - VoxCeleb1
  - VoxCeleb2
  - Speakers in the wild

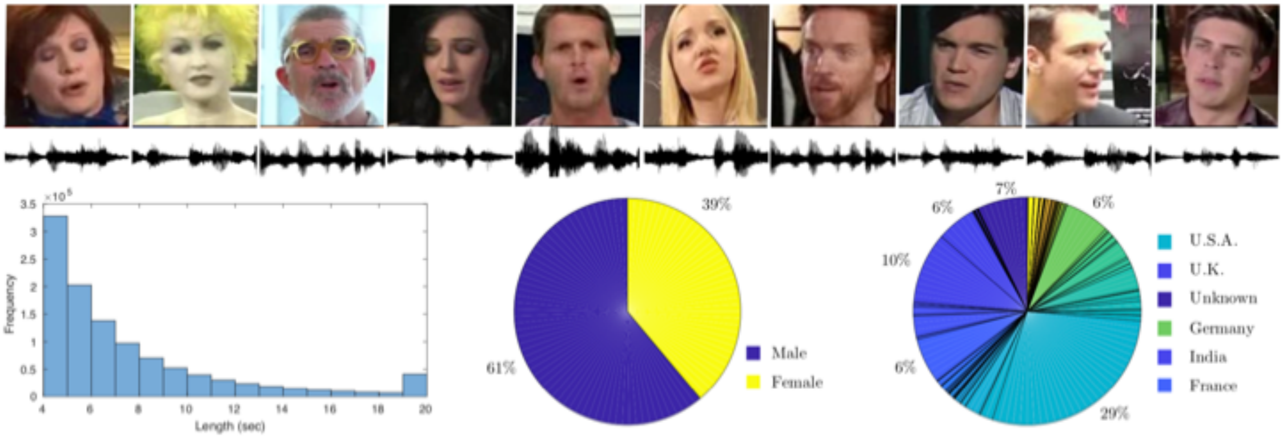


Figure from https://arxiv.org/abs/1806.05622

Sources Of Variability

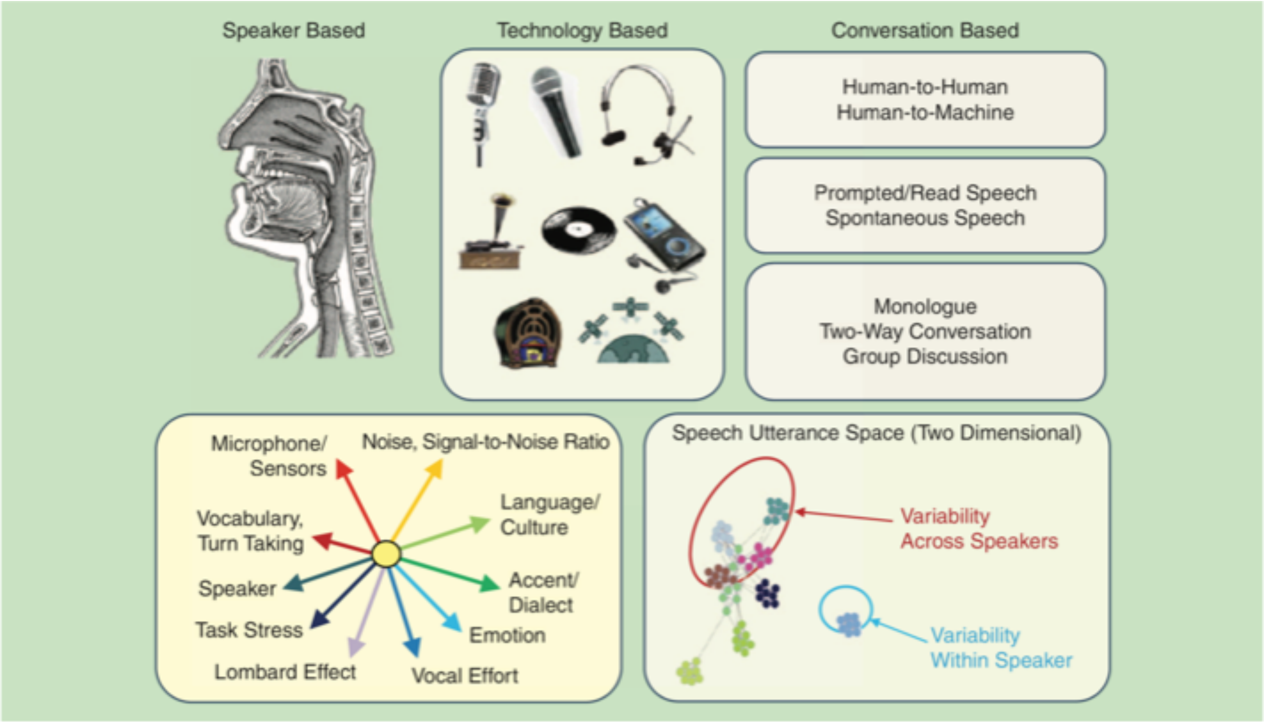


Figure from https://ieeexplore.ieee.org/document/7298570

More Challenges

- Text independent (vs text dependent)
- Different lengths of text
  - Can probably handle with pooling in the time dimension
- Scaling to a large number of people
  - Straightforward method is a 1 hot classification problem
  - But there's a point when this vector becomes too big
  - So need to think of a good strategy to handle
    - Possibly at this point better to project a speaker to a vector, then find the closest reference vector
    - Or use a hierarchical network head

For a comprehensive review see:

- Machine learning for speaker recognition
  http://www.eie.polyu.edu.hk/~mwmak/papers/IS2016-tutorial.pdf

Figure from add

A Basic Framework For Verification

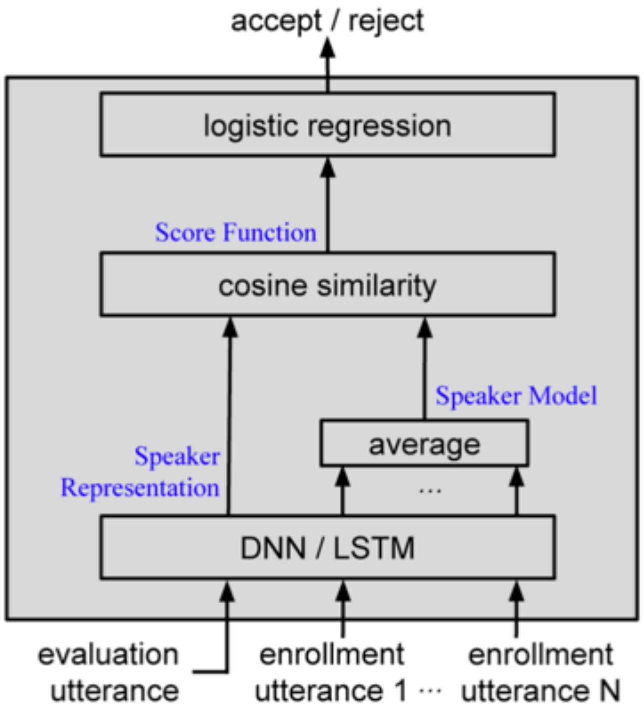
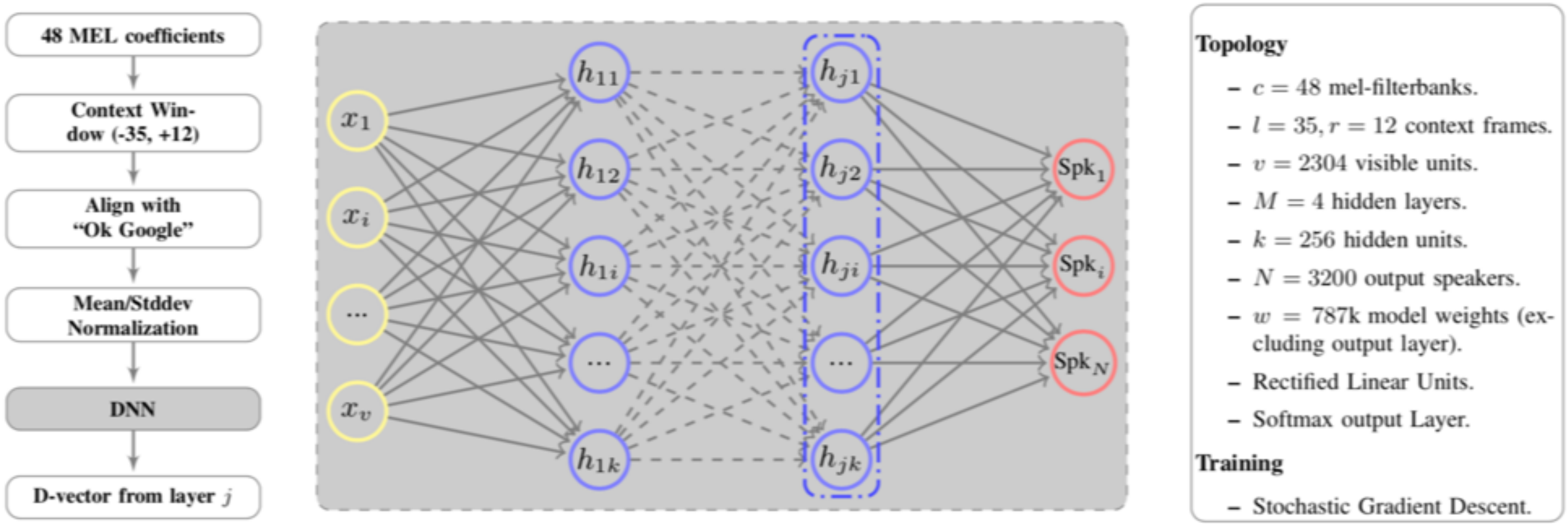


Figure from https://arxiv.org/abs/1509.08062

Small Footprint Speaker Recognition

Verification is based on cos distance of d vectors between enrolled and evaluation; $d_i = \max_t (h_{ji}^t)$



Note that this NN could be replaced with other xNN designs

Figure from https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43970.pdf

Example: CT-DNN

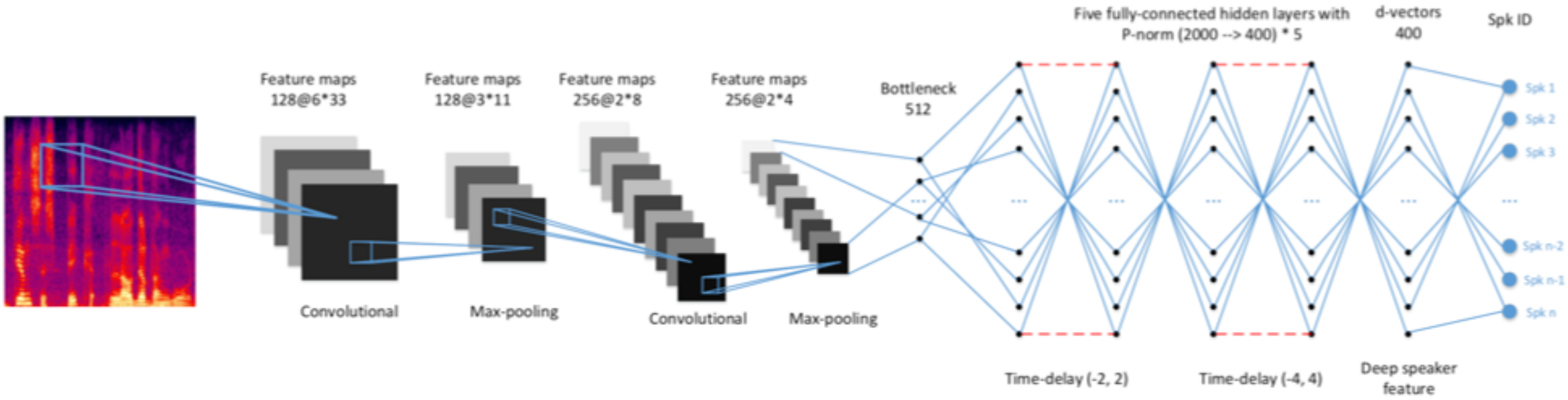


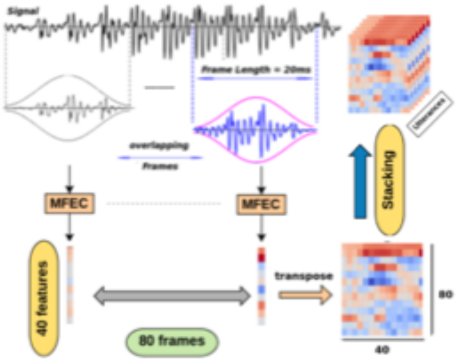
Figure from https://arxiv.org/abs/1705.03670

Example: VGGVox

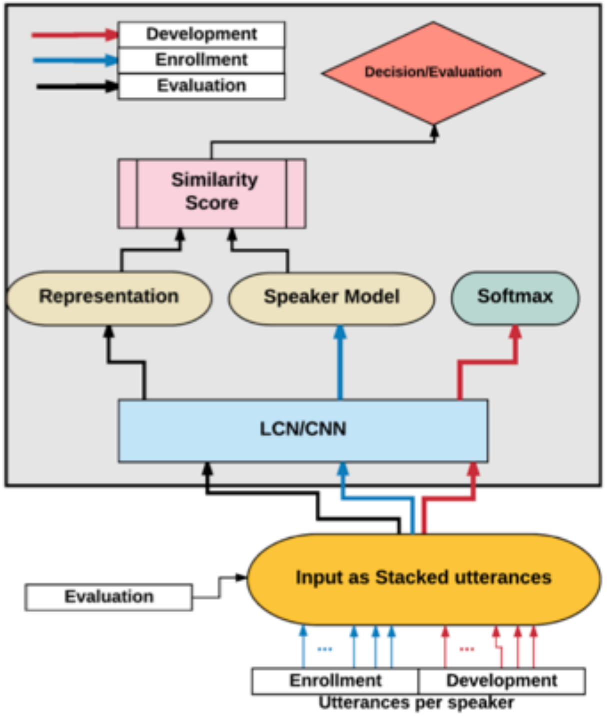
layer name	res-34	res-50
conv1 pool1	$7 \times 7, 64$, stride 2 3×3, max pool, stride 2	$7 \times 7, 64$, stride 2 3×3, max pool, stride 2
conv2_x	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4_x	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5_x	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
fc1 pool_time fc2	9×1, 512, stride 1 $1 \times N$, avg pool, stride 1 1×1, 5994	9×1, 2048, stride 1 $1 \times N$, avg pool, stride 1 1×1, 5994

Table from https://arxiv.org/abs/1806.05622

Example: 3D-CNN



layer	input-size	output-size	kernel	stride
Conv1-1	$\zeta \times 80 \times 40$	$80 \times 36 \times 16$	$3 \times 1 \times 5$	$1 \times 1 \times 1$
Conv1-2	$80 \times 36 \times 16$	$36 \times 36 \times 16$	$3 \times 9 \times 1$	$1 \times 2 \times 1$
Pool1	$36 \times 36 \times 16$	$36 \times 18 \times 16$	$1 \times 1 \times 2$	$1 \times 1 \times 2$
Conv2-1	$36 \times 18 \times 16$	$36 \times 15 \times 32$	$3 \times 1 \times 4$	$1 \times 1 \times 1$
Conv2-2	$36 \times 15 \times 32$	$15 \times 15 \times 32$	$3 \times 8 \times 1$	$1 \times 2 \times 1$
Pool2	$15 \times 15 \times 32$	$15 \times 7 \times 32$	$1 \times 1 \times 2$	$1 \times 1 \times 2$
Conv3-1	$15 \times 7 \times 32$	$15 \times 5 \times 64$	$3 \times 1 \times 3$	$1 \times 1 \times 1$
Conv3-2	$15 \times 5 \times 64$	$9 \times 5 \times 64$	$3 \times 7 \times 1$	$1 \times 1 \times 1$
Conv4-1	$9 \times 5 \times 64$	$9 \times 3 \times 128$	$3 \times 1 \times 3$	$1 \times 1 \times 1$
Conv4-2	$9 \times 3 \times 128$	$3 \times 3 \times 128$	$3 \times 7 \times 1$	$1 \times 1 \times 1$
FC5	$4 \times 3 \times 3 \times 128$	128	-	-



Figures from https://arxiv.org/abs/1705.09422

Keyword Spotting

Wake Up

- In a power constrained environment the wakeup process can be staged with less reliable lower power / complexity operations gating more reliable higher power / complexity operations

- Example flow
  - Sound detection
  - Voice activity detection
  - Wake word or phrase detection
  - Key word spotting or speech recognition

- These are all classification problems

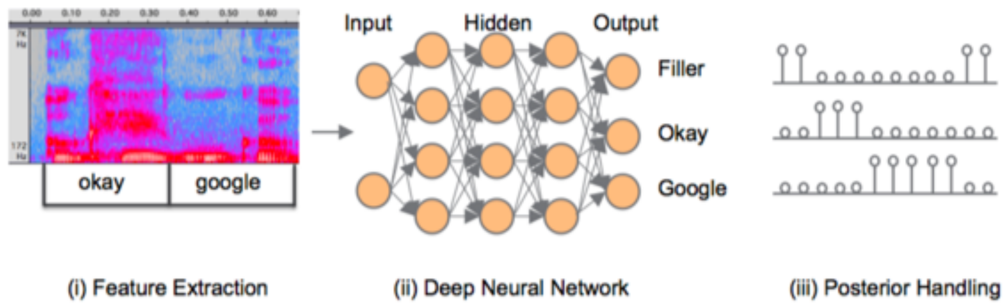


Figure from https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42537.pdf

Limited Vocabulary Speech Recognition

- Also referred to as keyword spotting or command recognition
- Like wake up but
  - A few more positive classes
  - Typically a little extra system power to work with
- Examples
  - Digits: Free spoken digit dataset (FSDD)
  - Commands: Google command data set

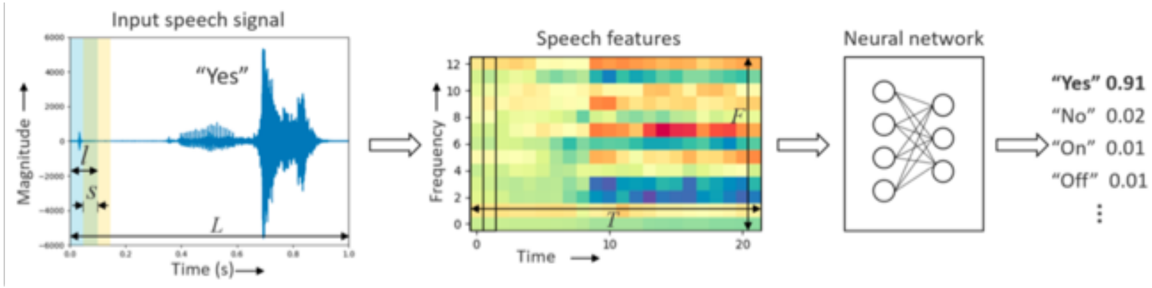
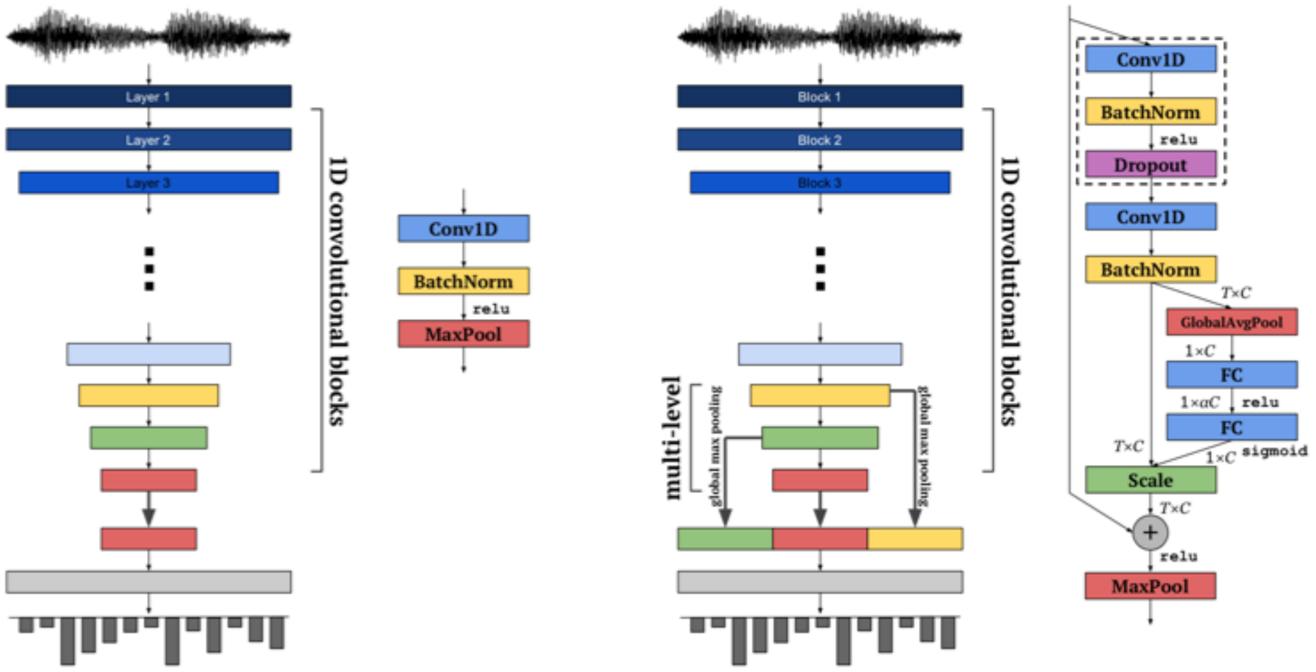


Figure from https://arxiv.org/abs/1711.07128

Raw Waveform Audio Classification

Residual network with squeeze and excite and multi level feature concatenation



(a) SampleCNN (b) ReSE-2-Multi

Figure from https://arxiv.org/abs/1712.00866

Confusion Matrix

A convenient way to visualize classification network performance; sometimes probabilities are intensity or color coded to make the visualization easier

Word spoken

Word recognized

<table>
<tr><th></th><th>w1</th><th>w2</th><th></th><th>wN</th></tr>
<tr><td>w1</td><td>P(w1 | w1)</td><td>P(w1 | w2)</td><td>⋯</td><td>P(w1 | wN)</td></tr>
<tr><td>w2</td><td>P(w2 | w1)</td><td>P(w2 | w2)</td><td>⋯</td><td>P(w2 | wN)</td></tr>
<tr><td></td><td>⋮</td><td>⋮</td><td>⋱</td><td>⋮</td></tr>
<tr><td>wN</td><td>P(wN | w1)</td><td>P(wN | w2)</td><td>⋯</td><td>P(wN | wN)</td></tr>
</table>

Conditional Modeling

Model → Conditional Model

A single framework for thinking about speech to text, text to speech, language translation and other problems with an input and feedback in the prediction

- Model
  - Assigns probabilities to a sequence of elements y_i (words, audio samples, ...)
    - $P(y_0, y_1, \ldots, y_{n-2}, y_{n-1}) = P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0) \ldots P(y_1 \mid y_0)\, P(y_0)$
  - So the key to creating a model is next element prediction (next word, next audio sample, ...) given previous elements
    - $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0)$

- It’s possible to cast a large number of problems as leaning a model using output data (next element prediction) then focusing / biasing the prediction by conditioning the model on input data **x**
  - $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0, \mathbf{x})$
  - Effectively, conditioning on the input data makes the next element prediction less uniform / more spikey (ideally 1 hot like)
  - Reduces the entropy of the conditional pmf
  - Needs input output data pairs for training

- During testing previous true outputs $y_{n-2}, \ldots, y_1, y_0$ are replaced with previous predicted outputs
  - $P(y_{n-1} \mid y_{n-2}^{hat}, \ldots, y_1^{hat}, y_0^{hat}, \mathbf{x})$
  - Use beam search or some similar variant to reduce error feedback
  - Even better if **x** contributes strongly to the prediction as that helps prevent error feedback effects

Conditional Model For Speech To Text

This is covered in the next section

- Learn a model for text that can predict the next phoneme, grapheme / character, word piece or word given previous phonemes, graphemes / characters, word pieces or words
  - This model can be optimized for specific or general types of text by training on that type of text
- Then focus / bias the text predictions via conditioning on features generated from a specific speech signal
  - This creates a conditional model that produces text corresponding to the given speech signal
- In equations (the network approximates this pmf)
  - Notation: y_i is the text; $\mathbf{x}$ is the speech signal
  - Text model: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0)$
  - Conditioned on features from speech: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0, \mathbf{x})$
  - Using previous predicted outputs: $P(y_{n-1} \mid y_{n-2}^{hat}, \ldots, y_1^{hat}, y_0^{hat}, \mathbf{x})$

Conditional Model For Text To Audio

This is covered in the section after the next section

- Learn a model for audio that can predict the next audio sample given previous audio samples
  - This model can be optimized for specific or general types of audio by training on that type of audio
  - Ex: human speech, 1980s hairspray metal, ...

- Then focus / bias the audio sample predictions via conditioning on features generated from a specific text
  - This creates a conditional model that produces audio (speech, music, ...) corresponding to the given text (words, instruments, ...)
  - Note that it's possible to condition on more than 1 thing (e.g., words + voice characteristics from a specific speaker to create a conditional model that produces speech corresponding to the specific words in the voice of the specific speaker)

- In equations (the network approximates this pmf)
  - Notation: y_i is the audio samples
$\mathbf{x}$ is the text signal
  - Audio model: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0)$
  - Conditioned on features from text: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0, \mathbf{x})$
  - Using previous predicted outputs: $P(y_{n-1} \mid y_{n-2}^{hat}, \ldots, y_1^{hat}, y_0^{hat}, \mathbf{x})$

Conditional Model For Language Translation

This is covered in the next lecture

- Learn a model for language 2 that can predict the next word in language 2 given previous words in language 2
  - This model can be optimized for specific or general types of text from language 2 by training on that type of text from language 2

- Then focus / bias the language 2 predictions via conditioning on features generated from specific text of language 1
  - This creates a conditional model that produces text of language 2 corresponding to the given text of language 1

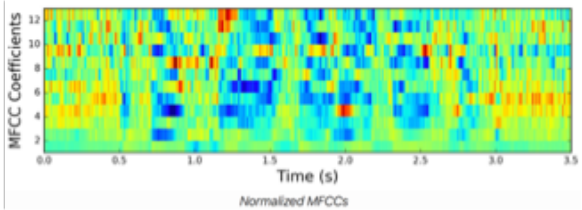
- In equations (the network approximates this pmf)
  - Notation: y_i are the words in language 2; $\mathbf{x}$ are the text features in language 1
  - Language 2 model: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0)$
  - Conditioned on features from language 1: $P(y_{n-1} \mid y_{n-2}, \ldots, y_1, y_0, \mathbf{x})$
  - Using previous predicted outputs: $P(y_{n-1} \mid y_{n-2}^{hat}, \ldots, y_1^{hat}, y_0^{hat}, \mathbf{x})$

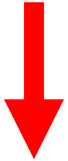
Speech To Text

Goal And Challenges

Also called speech recognition or automatic speech recognition

- Map audio inputs to linguistic outputs
  - This is a mapping from 1 sequence (e.g., sound) to another sequence (e.g., text); converting from 1 form to another is called transduction
- Challenge 1: alignment
  - Training data is typically in the form of an audio clip and a text transcription
  - What's not included is the alignment between the input audio and the associated text
  - What parts of the audio signal are useful for predicting the next output?
- Challenge 2: language
  - Want to ultimately predict words or sentences
  - But that space is too large to directly predict
  - To keep the size of the prediction space manageable it's common to predict phonemes, graphemes / characters or word pieces
  - These need to be composed into appropriate words and sentences
  - There's an underlying sequential structure that matters
  - Is it possible to exploit the structure of language?





The quick brown fox jumps over the lazy dog

Figure from https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

Data

- DeepSpeech
- Google Voice Search
- LibriSpeech
- The LJ speech dataset
- MGB
- Switchboard
- TED-LIUM
- Timit
- Wall Street Journal
- YouTube

Abbreviated History

- Classical (–ish): HMM–GMM
  - Predict phonemes (sounds) from speech via an acoustic model
  - Concatenate phonemes to words via a pronunciation model
  - Rescore sequences of words via a language model

- An intermediate hybrid step: HMM–xNN
  - Replace the GMM used for acoustic modeling with an xNN

- Present (or more precisely what's described here): xNN
  - The methods described here can still predict phonemes
  - But the focus will typically be on predicting graphemes (~ characters) from speech
    - Sometimes there's a 1 to 1 correspondence between phonemes and graphemes, sometimes a phoneme corresponds to multiple graphemes, sometimes a grapheme is unvoiced
    - Nice because a separate pronunciation model is no longer needed and a network can learn a basic pronunciation and a language model
  - Sometimes word pieces are predicted
    - Include graphemes as a subset of the word pieces so no out of model words

Sequence To Sequence Models

Here sequence 1 == sound and sequence 2 == words

- Pre processing to generate MFCCs or a variant is common
  - Other options are also possible
  - May 0 pad the input to a constant length

- Encoder generates strong features from MFCCs
  - RNN, LSTM, bi directional, pyramidal, CNN, ...
  - Possibly uses transfer learning for low resource languages

Think about the mapping required from the encoder (e.g., MFCCs to features corresponding to graphemes) – what is the optimal network structure for this?

- Decoder does alignment and phoneme, grapheme (character) or word piece pmf prediction and assembly
  - If doing phoneme prediction then a separate pronunciation model is required to generate words
  - If doing word piece prediction then typically include graphemes as a subset such that there are no out of vocabulary words
  - Possible to keep most likely predictions or a N best list of predictions which enables an external language model

- Post processing
  - Incorporation of an external language model to go from N best list to transcription

CTC

Connectionist temporal classification makes it possible to train xNNs with unknown input / output alignments (as typically found in speech training data)

- Note
  - CTC is described here in the context of grapheme prediction
  - Phoneme, word piece, ... are also possible

- Challenge 1: alignment
  - A xNN is a classifier that wants to map an input frame to a class
  - But the correct frame to class mapping is not known until the classifier is trained (because alignment is not given in training data)
  - This results in a circular dependency

- Solution: CTC loss function
  - The basic idea is for each frame for the xNN to predict a pmf over all graphemes plus a blank symbol
  - Repeated graphemes and the blank symbol are removed to create labels
  - The likelihood of each label is computed which allows the definition of a loss

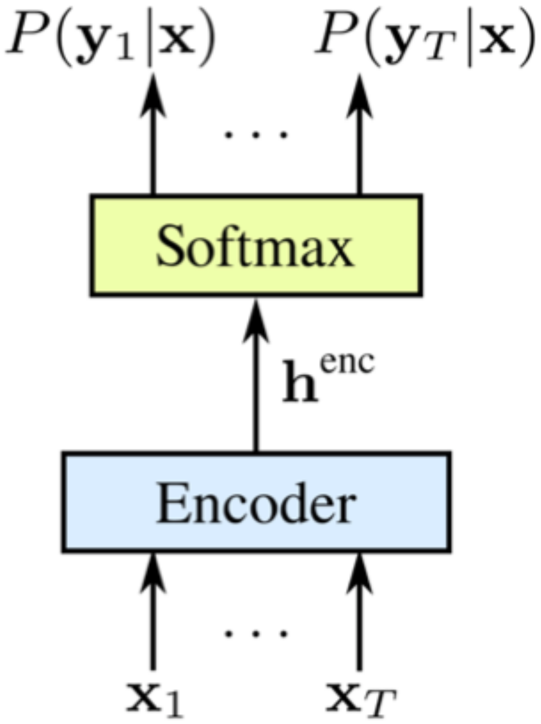


Figure from https://www.isca-speech.org/archive/Interspeech_2017/pdfs/0233.PDF

CTC Decoding

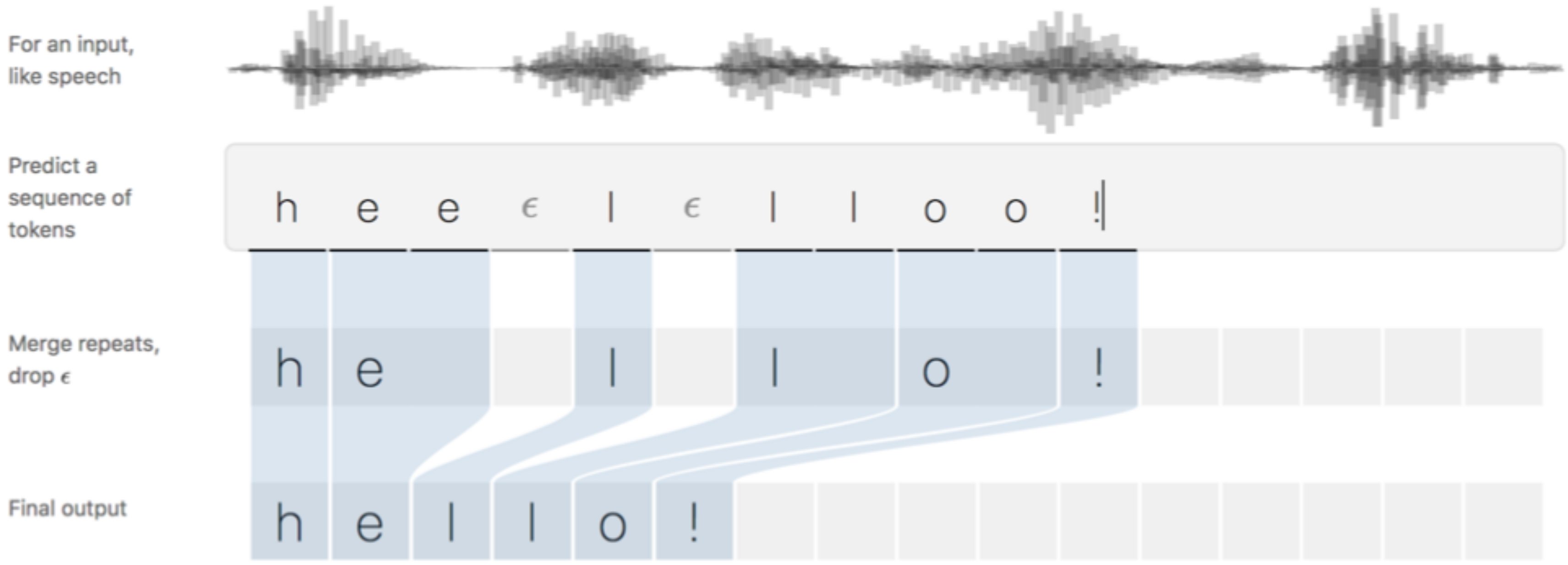


Figure from https://distill.pub/2017/ctc/

CTC Decoding

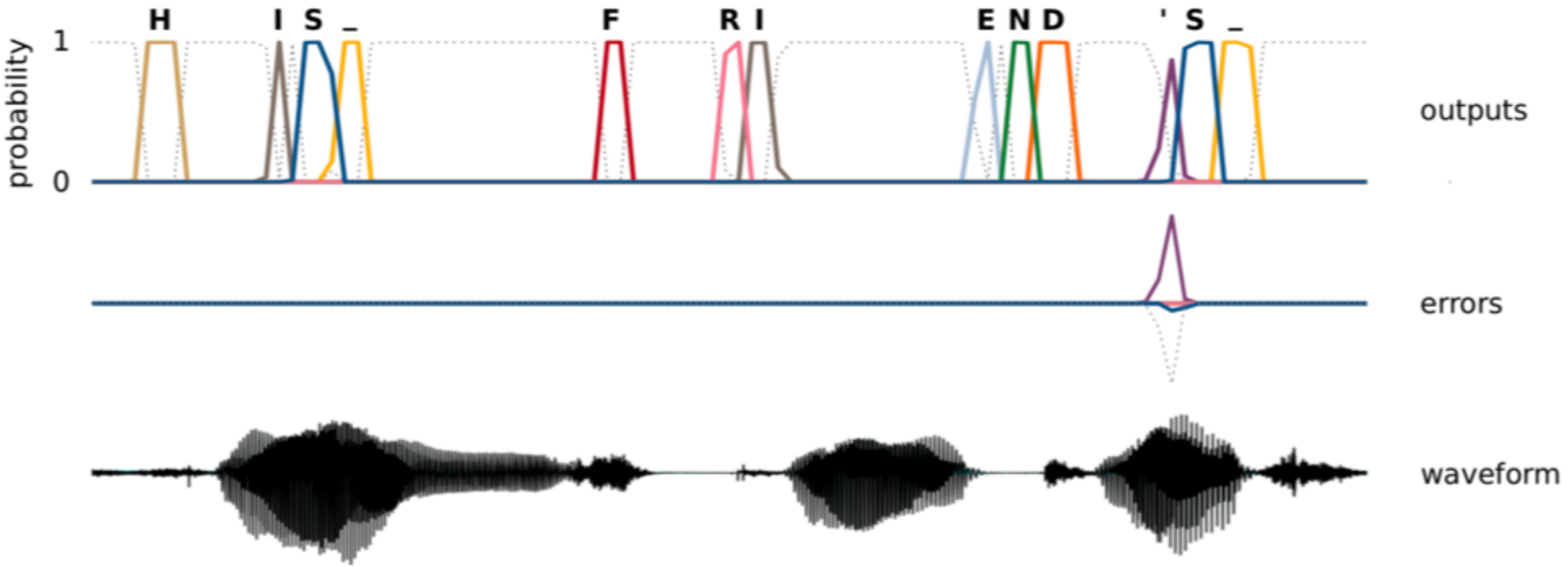


Figure from http://proceedings.mlr.press/v32/graves14.pdf

CTC Definitions

- X is the network input (e.g., bins x T’)
- Y is the network output (e.g., classes x T)
  - Let classes == {a, ..., z, –}
  - So for each time frame the network is predicting a pmf over all the letters + blank (the ‘–’) for the corresponding input frame
  - Note that other options are possible for the classes (e.g., phonemes for smaller, word pieces for larger)
  - Note that the T’ can be the same size as T or different (frequently T is smaller than T’)
  - The key is the inclusion of the blank class for alignment purposes
- π is a path that corresponds to selecting 1 class at each of the T times
  - Ex: π = (a, a, a, –, r, –, –, t, t, t, t, –)

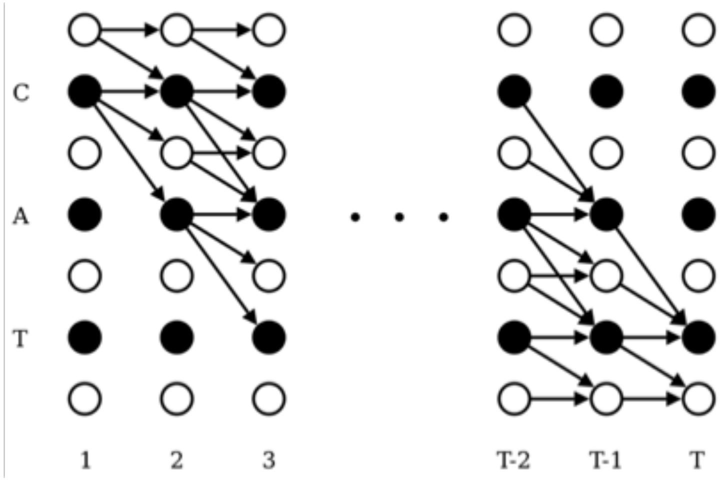


Figure from https://www.cs.toronto.edu/~graves/icml_2006.pdf

CTC Definitions

- B is a many to 1 mapping that takes a path π and 1st removes repeated classes then 2nd removes blanks to produce a label ℓ
  - Ex: $\ell = B(\pi) = B(a, a, a, –, r, –, –, t, t, t, t, –) = (a, r, t)$
  - Note that this ordering of what is removed allows repeated letters to potentially be produced such as the 'tt' in 'letter'

- ℓ is a label that corresponds to (potentially exponentially) many paths
  - Ex: The set of all path corresponding to a label
    $\{\pi\} = B^{-1}(\ell) = B^{-1}(a, r, t) =$
    {(a, r, t, –, –, –, –, –, –, –, –, –),
    (a, a, r, t, –, –, –, –, –, –, –, –),
    (a, a, –, r, t, –, –, –, –, –, –, –),
    ...}

CTC Output

- The probability of a path π given input X
  - $P(\pi \mid X) = \prod_t Y(\pi(t), t)$
  - The product of the pmf values of the class elements for the path
- The probability of a label ℓ given input X
  - $P(\ell \mid X) = \Sigma\, P(\pi_i \mid X)\ \forall\ \pi_i \in B^{-1}(\ell)$
  - This is a sum over all paths that can possibly generate the label
  - But in practice only a small number of paths will contribute meaningfully to the probability (multiplying multiple small numbers corresponding to unlikely classes results in a very small value)
  - This observation can be used to simplify the probability calculation via dynamic programming
- Training vs testing
  - During training, parameters are optimized to minimize the negative log likelihood of $P(\ell \mid X)$
  - During testing, either greedy / arg max decoding or beam search can be used (more on beam search in a few slides)

CTC Properties

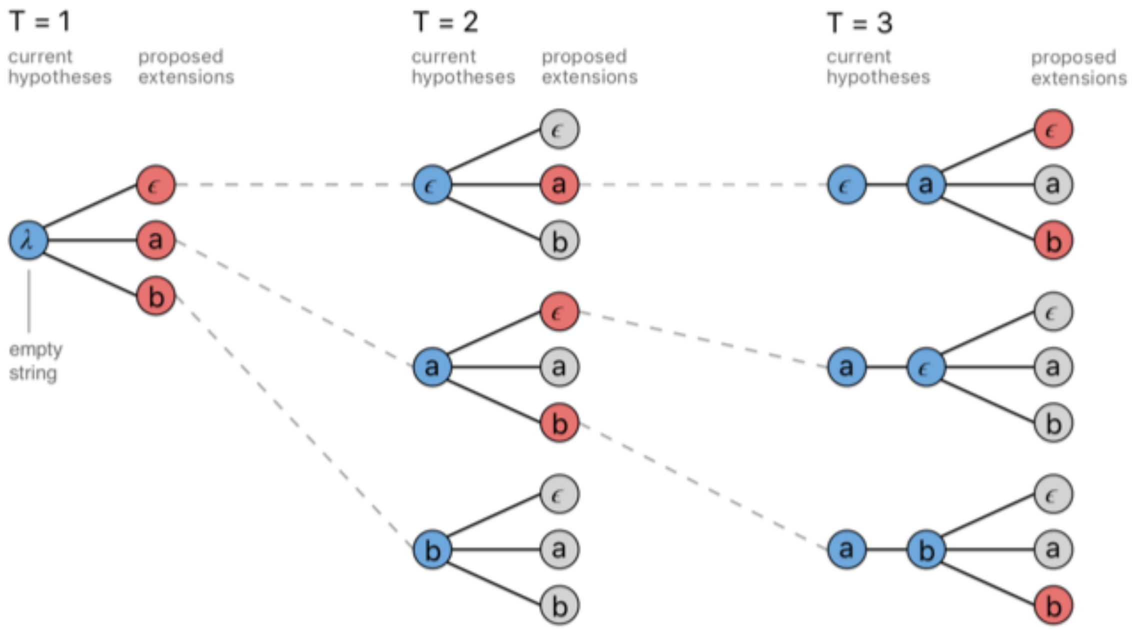
- The specific network used to map from input features to output classes used in the original paper was a bidirectional LSTM
  - A nice part of this is the forwards and backwards sequential combination of information to map from weak features (MFCC) to classes
  - However, the use of this network is not a requirement and other networks can be used
- Network output predictions are independent
  - This does not take advantage of the structure of language (challenge 2)
  - This can be addressed in multiple ways, the most common being beam search decoding to a N best list and the inclusion of an external language model(s)
  - This makes adaptation to new domains easier as only the external language model changes

CTC Properties

- The output sequence length cannot be longer than the input sequence length
  - This is not a big deal for speech to text as the number of speech frames is typically $\gg$ the output graphemes
  - But it will be an issue for using CTC in text to speech systems
- Alignments are monotonic from output classes to labels
  - This is reasonable for speech
  - But not valid for other sequence to sequence models like language translation

Beam Search

A common method for going for a sequence of grapheme probabilities to letters; beam size = 1 is greedy decoding

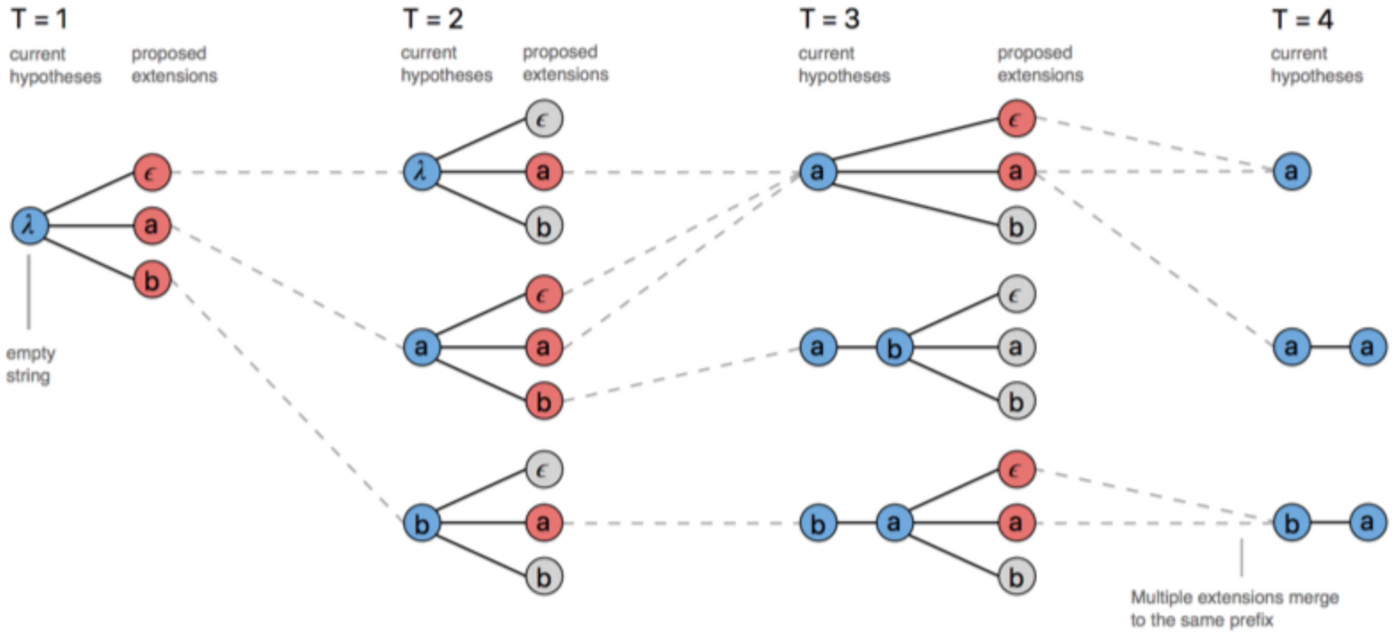


A standard beam search algorithm with an alphabet of $\{\epsilon, a, b\}$ and a beam size of three.

Figure from https://distill.pub/2017/ctc/

Beam Search

The CTC blank symbol introduces a few challenges into the standard beam search algorithm



The CTC beam search algorithm with an output alphabet $\{\epsilon, a, b\}$ and a beam size of three.

Figure from https://distill.pub/2017/ctc/

Beam Search

The CTC blank symbol introduces a few challenges into the standard beam search algorithm

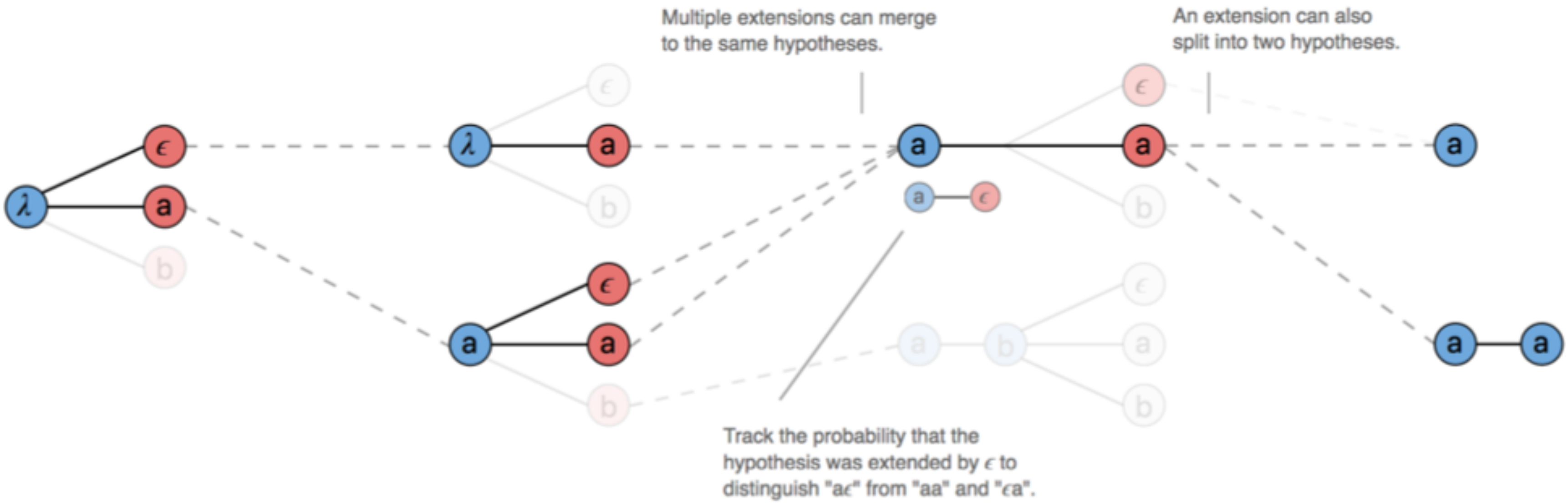


Figure from https://distill.pub/2017/ctc/

External Language Model

The language lecture will look at this in more detail

- A language model assigns a probability to a sequence of words
  - $P(w_0, w_1, \ldots, w_{n-2}, w_{n-1}) = P(w_0)\, P(w_1 \mid w_0) \ldots P(w_{n-1} \mid w_{n-2}, \ldots, w_1, w_0)$
  - So it's possible to create from learning to predict the next word for all different length sequences $P(w_{n-1} \mid w_{n-2}, \ldots, w_1, w_0)$
  - The nice part about this is that it can be learned offline using very large sources of text
  - A few different options for this will be discussed in the language lecture

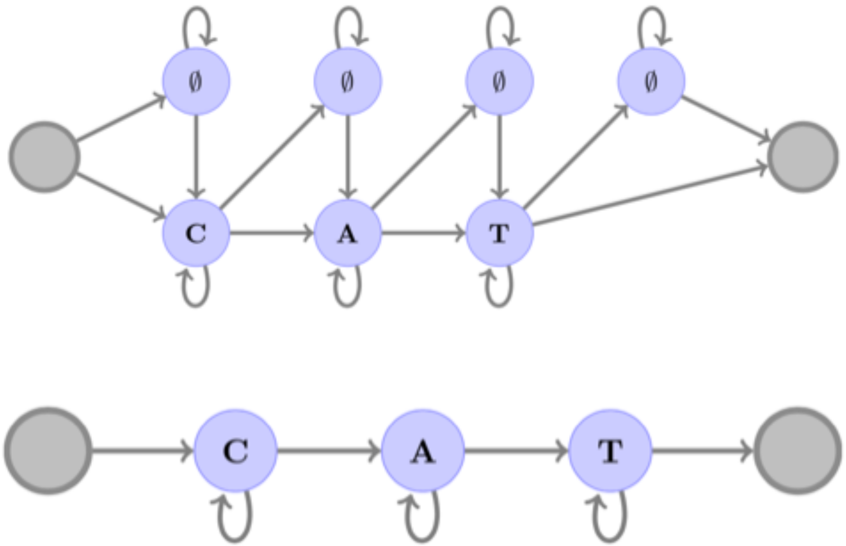
- N gram language model
  - Given N – 1 words, predict the pmf of the Nth word

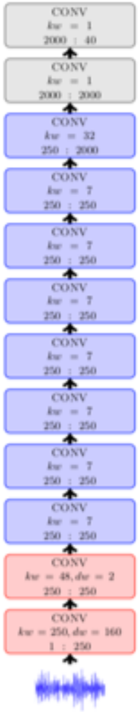
- How can this be used for speech recognition?
  - Incorporate the language model into beam search by complementing the pmf of letters / words produced by the network with a pmf of letters / words based on language in the form of a conditional language model
  - Use the language model after beam search decoding to re score / revise a B best list of transcriptions
  - Example uses in linear and log form
    - $\ell^* = \arg\max_{\ell} (P(\ell \mid X)\, P_{LM}(\ell)^{\alpha}\, \text{length}(\ell)^{\beta})$
    - $\ell^* = \arg\max_{\ell} (\log(P(\ell \mid X)) + \alpha \log(P_{LM}(\ell)) + \beta \log(\text{length}(\ell)))$

Auto Segmentation Criteria

A modification to the CRC criteria; CTC graph for ‘cat’ on top, AutoSeg graph on bottom; 1st network layer is only present when the input is a raw audio

- 3 types inputs were tested
  - MFCC
  - Power spectrum
  - Raw waveform
- Network (other options possible)
  - 1D CNNs
  - Atrous convolution
- Output (AutoSeg criteria)
  - Grapheme class pmf
  - No blank class
  - Add a 2x and 3x repetition class
  - Leads to simpler graphs
  - But potentially more complex to train





Figures from https://arxiv.org/abs/1609.03193

RNN Transducer

RNN transducer adds a data dependent language model that predicts the present letter given previous letters

- Basics
  - Uses the CTC strategy for alignment
  - Adds conditional dependence in the predicted graphemes by feeding back previous outputs to a prediction network for an implicit language model
    - True value during training
    - Most likely prediction during testing
    - This is an implicit language model
- Encoder and internal prediction network
  - As before, various xNN architectures are possible
  - Note that the internal prediction network needs to be optimized for the target usage

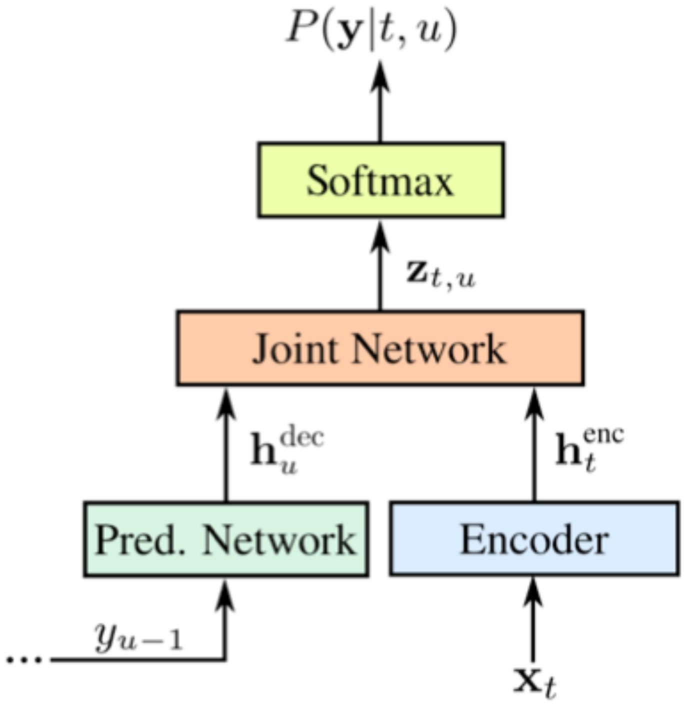


Figure from https://arxiv.org/abs/1801.00841

RNN Transducer

RNN transducer adds a data dependent language model that predicts the present letter given previous letters

- Equations
  - $h_{t,u}^{joint} = \tanh(A\ h_t^{enc} + B\ h_u^{dec} + c)$
  - $z_{t,u} = D\ h_{t,u}^{joint} + e$
  - $P(Y \mid t, u) = \text{softmax}(z_{t,u})$

- Output is a probability cube
  - Graphemes + blank x T frames x U outputs
  - Beam search style decoding can be used to convert to a sequence
  - Output sequence can be longer than the input sequence based on traversal method through the cube

- External language models can be used to augment the internal language model
  - Needs to be trained on a large separate text for maximum effectiveness

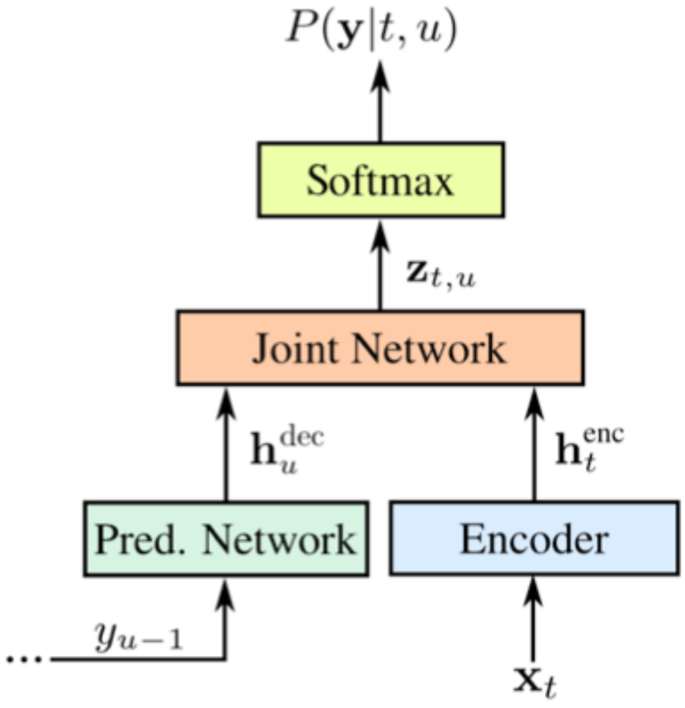
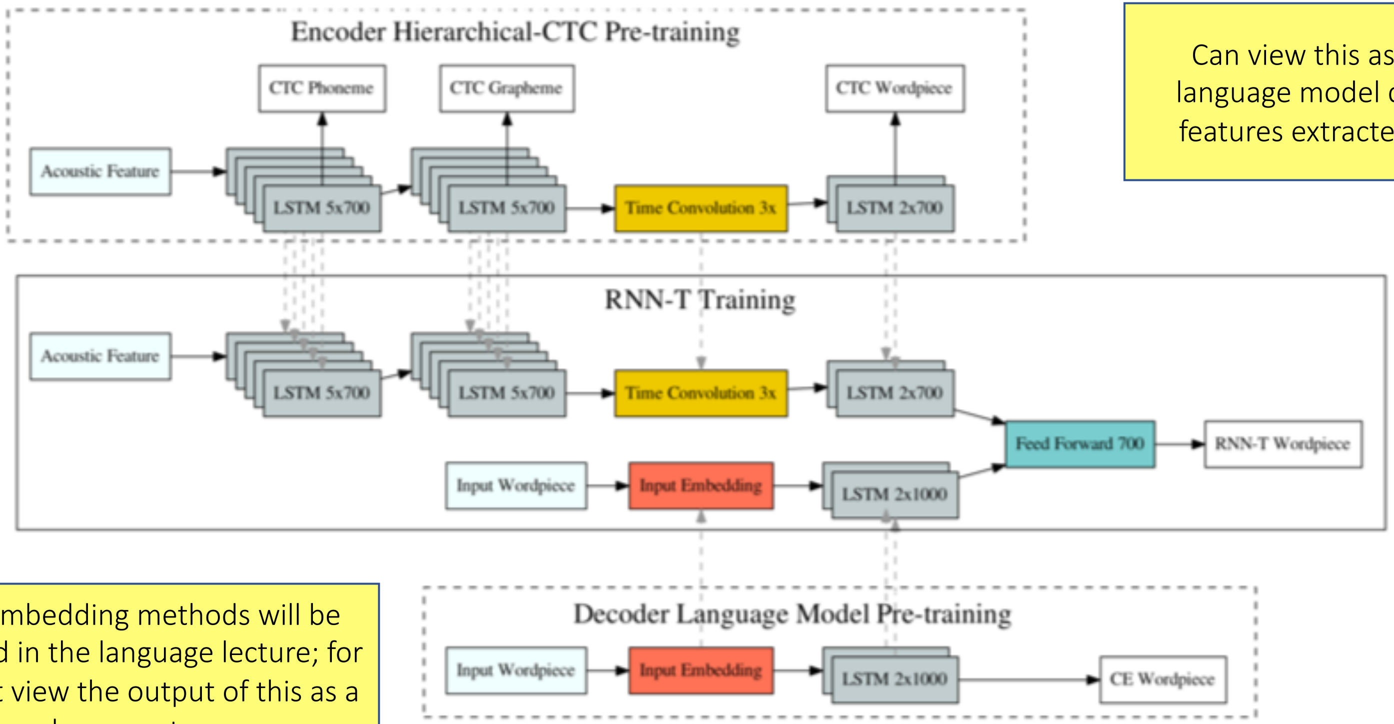


Figure from https://arxiv.org/abs/1801.00841

RNN Transducer

RNN transducer adds a data dependent language model that predicts the present letter given previous letters (or in this case word piece)



Can view this as a word piece language model conditioned on features extracted from speech

Input embedding methods will be discussed in the language lecture; for now, just view the output of this as a dense vector

Figure from https://arxiv.org/abs/1801.00841

Attention

- Attention
  - Sequence to sequence models encode a variable length input into hidden states
  - Attention allows a decoder to adaptively select hidden states to generate the output
- Attention is a general strategy that applies to more than just speech
  - Encoder, attender, decoder structure
  - Different combinations of inputs are used to create each output (i.e., the network learns what part of the input to pay attention to to generate each output)
  - Monotonic alignments are not required; this is especially beneficial for translation
- Example speech to text system: listen attend spell
  - Listen - encoder acoustic model generates features
  - Attend - attention creates a context vector from all the features appropriate for the decoder state
  - Spell - decoder outputs graphemes; the conditional dependence in graphemes create an implicit internal language model

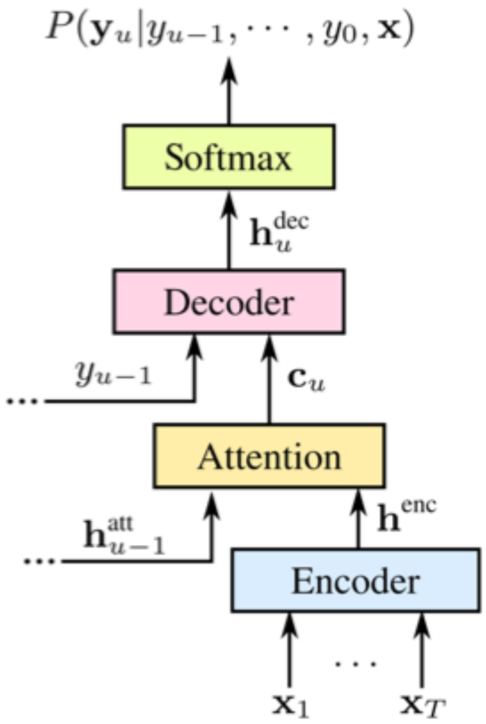


Figure from https://www.isca-speech.org/archive/Interspeech_2017/pdfs/0233.PDF

Attention

- Attention mechanism to create context vector
  - h_{u-1}^{att} is the state of the lowest layer of the decoder after predicting the previous symbol
  - $\beta_{t,u} = \langle \phi(h_t^{enc}), \psi(h_{u-1}^{att}) \rangle$, ϕ and ψ are learnable linear embeddings
  - $\alpha_{:,u} = \text{softmax}(\beta_{:,u})$
  - $c_u = \text{diag}(\alpha_{:,u})\ h_t^{enc}$

- Similar decoding options are available
  - Greedy
  - Beam
  - Beam + external language model
  - ...

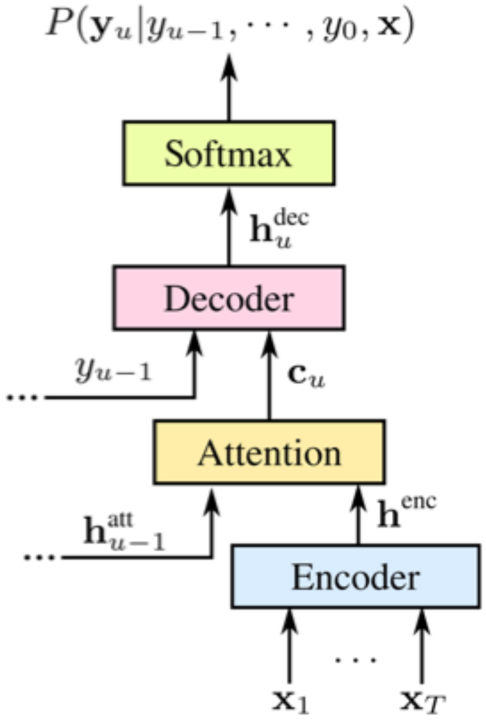


Figure from https://www.isca-speech.org/archive/Interspeech_2017/pdfs/0233.PDF

Comparing Transducer Transition Possibilities

Output symbol vs input frame transition possibilities for the word 'cat'

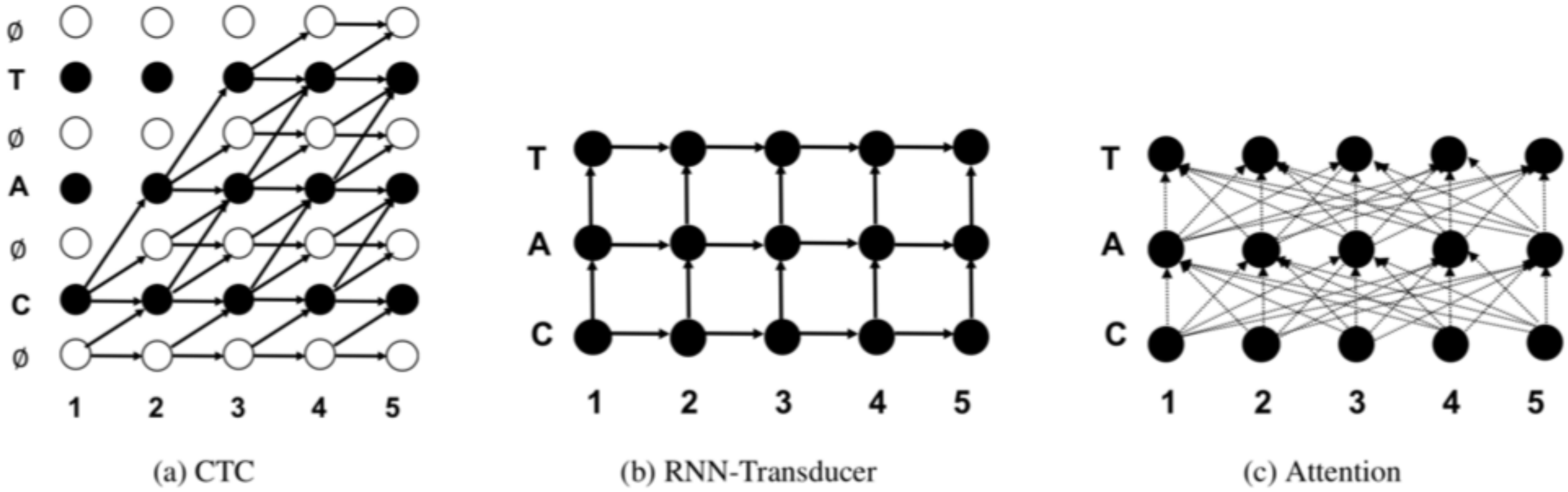


Figure from https://arxiv.org/pdf/1707.07413.pdf

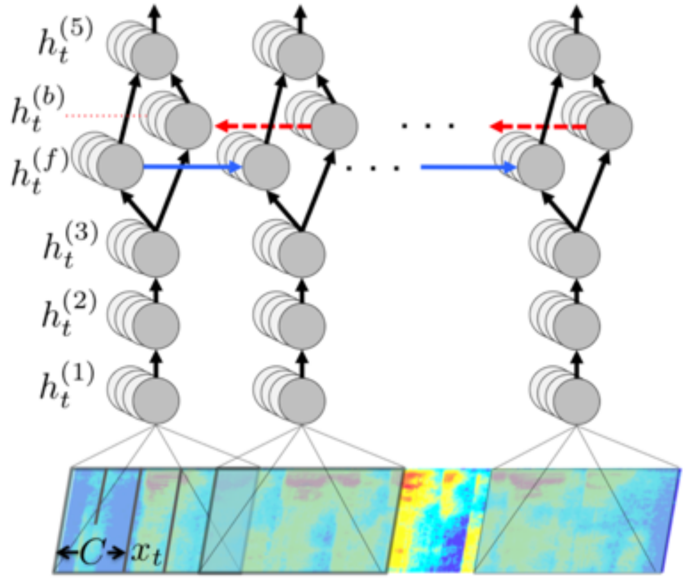
Comparing Transducers Alignments

Output symbol vs input frame alignment for the phrase ‘the order backlog is extremely strong well into next year’

Figure from https://arxiv.org/pdf/1707.07413.pdf

Example: Baidu's DeepSpeech

- DeepSpeech 1 (2014)
  - 7000 hours of labeled English speech + augmentation
  - A simple model is competitive with state of the art
  - Multilayer RNN encoder with a CTC loss function
- DeepSpeech 2 (2015)
  - Generalized to multiple languages
  - 11000+ hours of labeled English and 9000+ hours of labeled Mandarin + augmentation
  - 10s of exaflops for training
  - Multilayer convolution + RNN encoder with a CTC loss function
- DeepSpeech 3 (2017)
  - Switched from CTC to RNN transducer with an external pre trained language model used during training (see the blog post and ColdFusion paper)

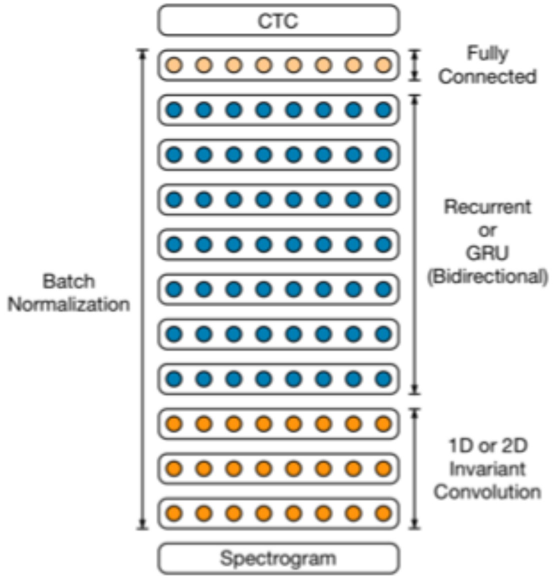


Note the equal focus on algorithms and software / hardware implementation

Figure from https://arxiv.org/pdf/1412.5567.pdf

Example: Baidu's DeepSpeech

- DeepSpeech 1 (2014)
  - 7000 hours of labeled English speech + augmentation
  - A simple model is competitive with state of the art
  - Multilayer RNN encoder with a CTC loss function
- DeepSpeech 2 (2015)
  - Generalized to multiple languages
  - 11000+ hours of labeled English and 9000+ hours of labeled Mandarin + augmentation
  - 10s of exaflops for training
  - Multilayer convolution + RNN encoder with a CTC loss function
- DeepSpeech 3 (2017)
  - Switched from CTC to RNN transducer with an external pre trained language model used during training (see the blog post and ColdFusion paper)



Note the equal focus on algorithms and software / hardware implementation

Figure from https://arxiv.org/pdf/1412.5567.pdf

Example: Google's Listen Attend Spell

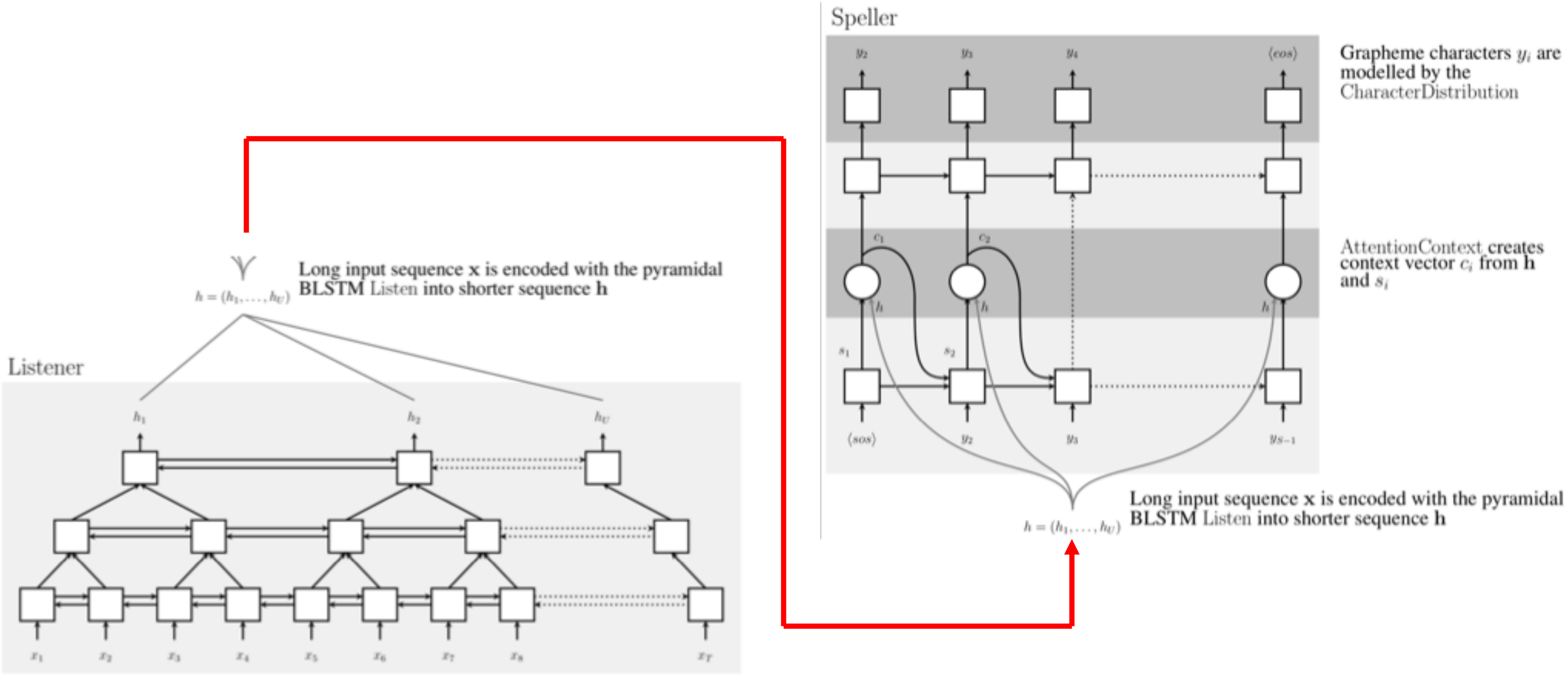


Figure from https://arxiv.org/abs/1508.01211

Example: Google's Listen Attend Spell

- Output: $P(y \mid x) = \prod_i P(y_i \mid x, y_{<i})$

- Listen: $h = \text{Listen}(x)$
  - Let i be the time step and j be the layer
  - 1x: $h_i^j = \text{BLSTM}(h_{i-1}^j, h_i^{j-1})$
  - 3x: $h_i^j = \text{pBLSTM}(h_{i-1}^j, [h_{2i}^{j-1}, h_{2i+1}^{j-1}])$

- Attend and spell: $P(y \mid x) = \text{AttendAndSpell}(h, y)$
  - $c_i = \text{AttentionContext}(s_i, h)$
    - See attention slide
  - $s_i = \text{RNN}(s_{i-1}, y_{i-1}, c_{i-1})$
    - RNN is a 2 layer LSTM
  - $P(y_i \mid x, y_{<i}) = \text{CharacterDistribution}(s_i, c_i)$
    - CharacterDistribution is a MLP with softmax output over characters

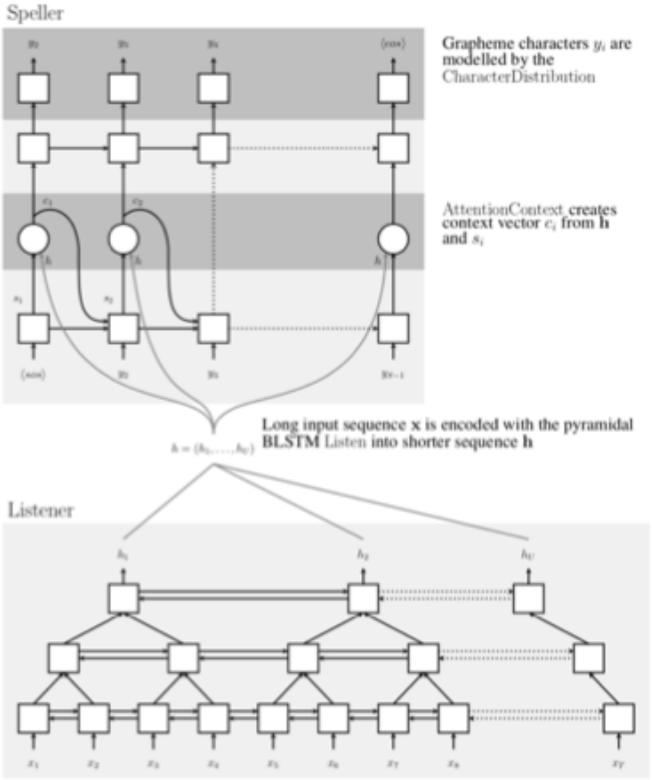


Figure from https://arxiv.org/abs/1508.01211

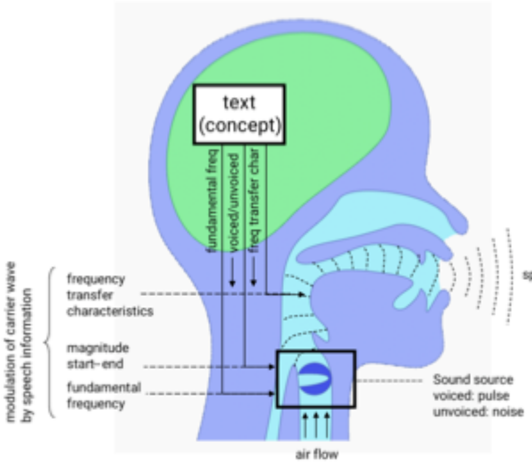
Evaluation

- Metrics
  - Word error rate
  - Substitutions
  - Insertions
  - Deletions
- Leaderboard (also a really good web address)
  - https://github.com/syhw/wer_are_we

Text To Speech

Goal

- Generate speech from text
  - Also called speech synthesis
- This is a sequence to sequence transduction problem just like speech to text (but now with a many to 1 mapping)
- History
  - Rule based formant synthesis
  - Sample based concatenative synthesis
  - Model based generative synthesis
- Here we'll mainly look at model based generative synthesis including sequence to sequence based methods



Figures from https://www.thoughtco.com/what-is-a-speak-and-spell-1992413 and https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%2010%20-%20Text%20to%20Speech.pdf

Basics

- Given
  - Training speech waveforms
  - Training text
  - Testing text
- Generate
  - Testing speech waveforms
- In a model based system
  - The training speech waveforms and training text are used to create a model
  - The model is used with the testing text to generate testing speech waveforms

For a nice introduction to text to speech methods with much more information than is included here, the following links are a good starting point:

- Generative model-based text-to-speech synthesis
  https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%2010%20-%20Text%20to%20Speech.pdf
- Text normalization, letter to sound, prosody
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec14.pdf
- Waveform synthesis in TTS
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec15.pdf
- Parametric TTS, intoxication, depression, trauma, personality
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec16.pdf

Basics

- 2 parts
  - Part 1: convert text into an intermediate representation; traditional steps include
    - Text normalization
    - Grapheme to phoneme conversion
    - Prosodic features generation
  - Part 2: convert the intermediate representation into audio

- Traditional part 1 is linguistic, duration and F_0 features

- Traditional part 2 is Griffin-Lim algorithm (if the output of part 1 is a spectrogram)

For a nice introduction to text to speech methods with much more information than is included here, the following links are a good starting point:

- Generative model-based text-to-speech synthesis
  https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%2010%20-%20Text%20to%20Speech.pdf
- Text normalization, letter to sound, prosody
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec14.pdf
- Waveform synthesis in TTS
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec15.pdf
- Parametric TTS, intoxication, depression, trauma, personality
  http://web.stanford.edu/class/cs224s/lectures/224s.17.lec16.pdf

Tacotron 1

An incomplete list of things I never expected in life: to present a slide with the title Tacotron at the top

- Neural attention based sequence to sequence model that predicts linear spectrograms from characters
  - Linear spectrograms are used as an intermediate representation

- Training input is text and audio pairs

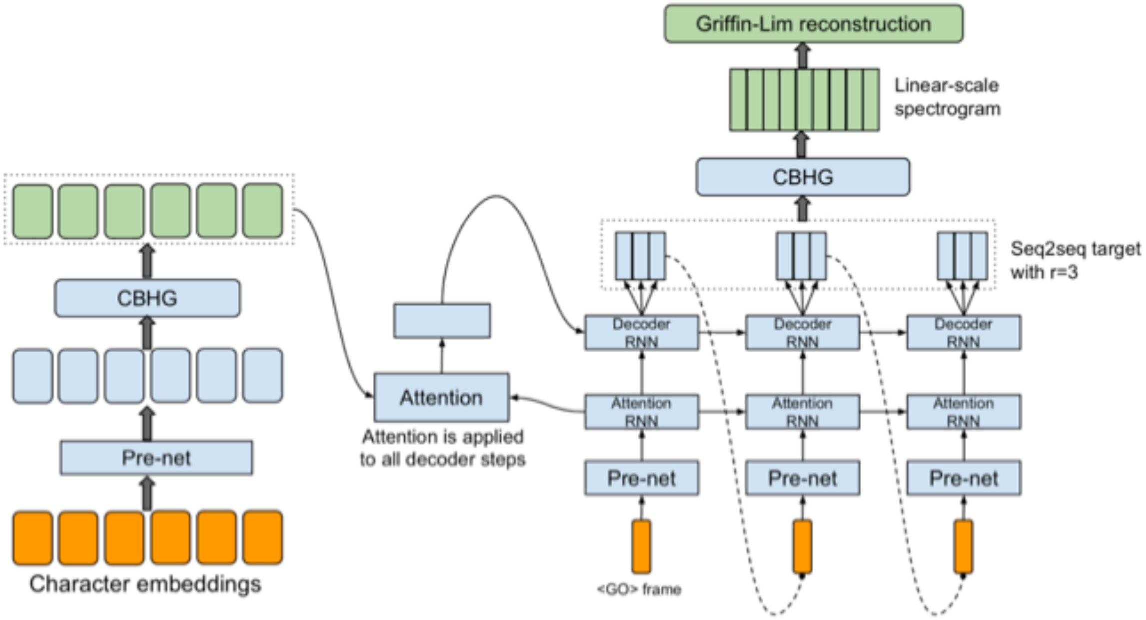


Figure from https://arxiv.org/abs/1703.10135

Tacotron 1

An incomplete list of things I never expected in life: to present a slide with the title Tacotron at the top

- 1 hot encoded characters are embedded into a vector
- The pre net is a bottleneck layer with dropout
- A CBHG network creates the encoder output features
  - 1D convolution
  - Highway network
  - Bi directional GRU

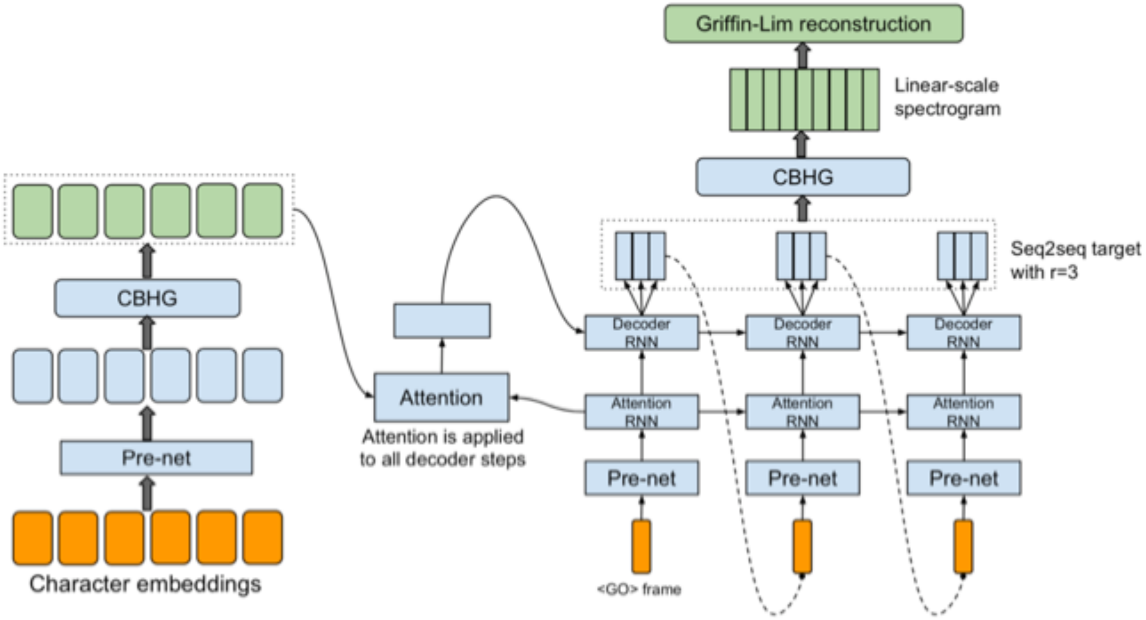


Figure from https://arxiv.org/abs/1703.10135

Tacotron 1

An incomplete list of things I never expected in life: to present a slide with the title Tacotron at the top

- Attention RNN produces an attention query and the resulting context vector is concatenated with with the attention RNN output
- The decoder RNN is a stack of GRUs with vertical residual connections
    - The decoder target is a 80 band mel spectrogram
    - Multiple non overlapping frames are predicted at each stage
    - The last predicted frame is fed back as the next input to the decoder

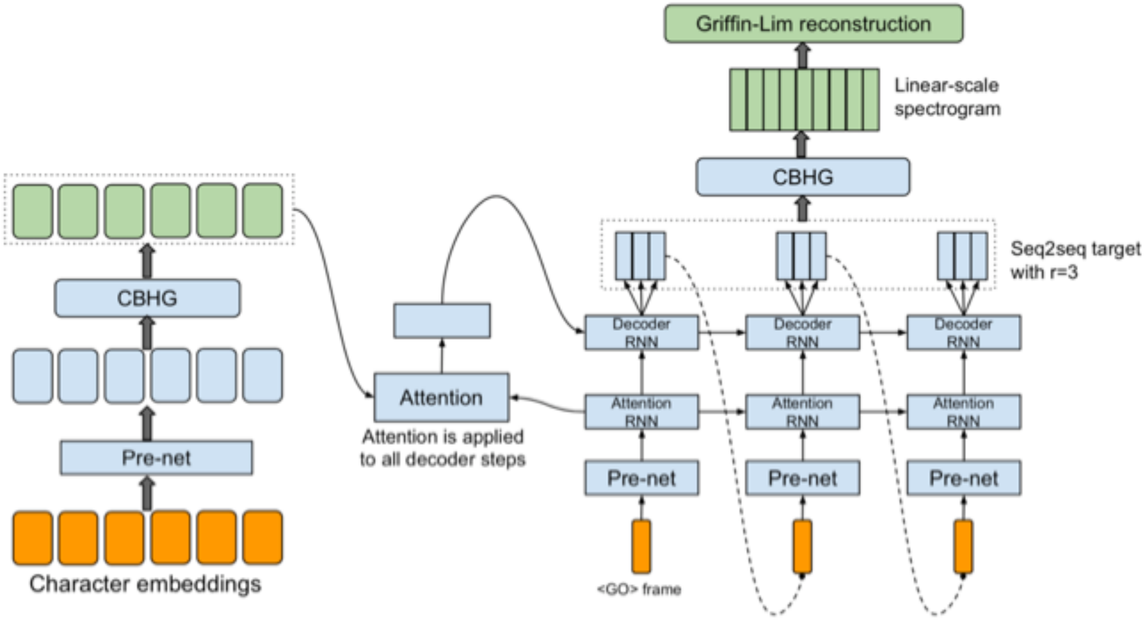


Figure from https://arxiv.org/abs/1703.10135

Tacotron 1

An incomplete list of things I never expected in life: to present a slide with the title Tacotron at the top

- The final CBHG network predicts linear scale spectrograms from the 80 band mel spectrogram

- Traditional Griffin-Lim reconstruction is used to go from linear spectrograms to time domain audio samples

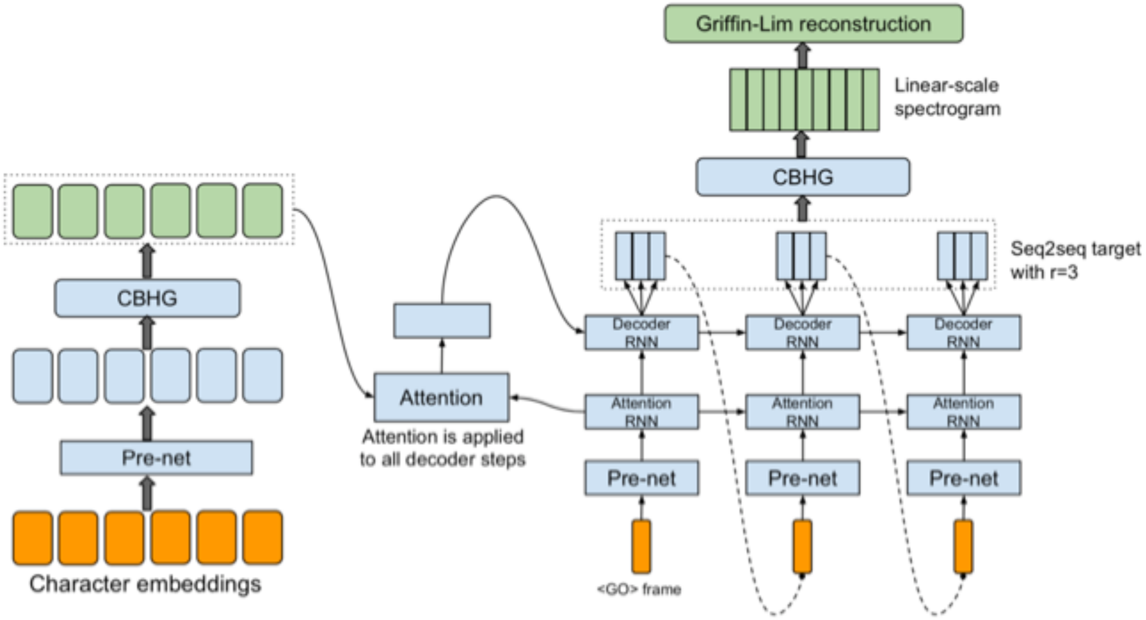


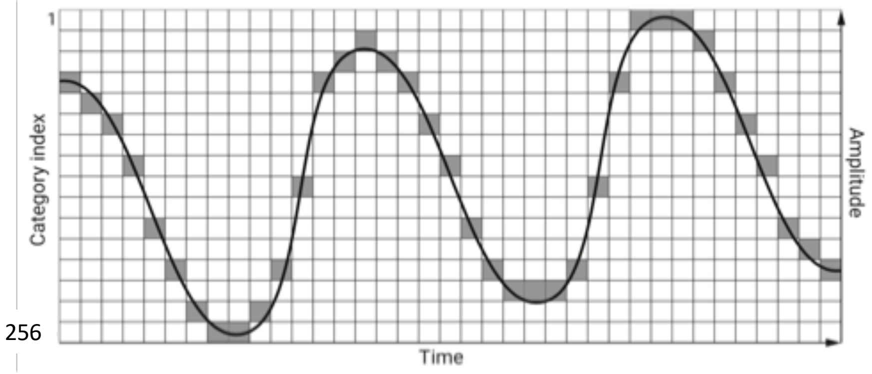
Figure from https://arxiv.org/abs/1703.10135

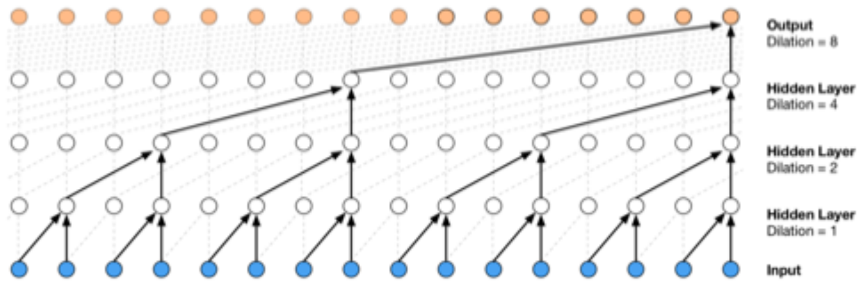
WaveNet

- Sample by sample classification to an output level that is conditioned on previous outputs, speaker and linguistic features of the text

- Output waveform x = {x_1, ..., x_T} factored as
  - $p(x) = \prod_t p(x_t \mid x_1, \ldots, x_{t-1})$
  - Each sample x_t is conditioned on all previous samples
  - During testing the previous output sample is fed back into the network to produce the next sample such that generation is sequential

- Causal (non bi directional convolution)
  - Dilated convolution to increase receptive field size
  - The dilation factor is doubled after every layer up to a limit of 512 then the process is repeated





Figures from https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%2010%20-%20Text%20to%20Speech.pdf and https://arxiv.org/abs/1609.03499

WaveNet

- Gated activation units
  - $z = \tanh(W_{f,k} \circledast x) \odot \sigma(W_{g,k} \circledast x)$
  - k is the layer index
  - $W_{f,k}$ is a learnable filter, $W_{g,k}$ is a learnable gate
  - Chosen because it worked better than ReLU in experiments

- Residual connections in the encoder with summed skip connections in the decoder

- μ law quantization x_t of to 256 levels
  - $f(x_t) = \text{sign}(x_t) \ln(1 + \mu |x_t|) / \ln(1 + \mu)$
  - $\mu = 256, -1 < x_t < 1$
  - Note that this is a non uniform quantization method that is well matched to human speech

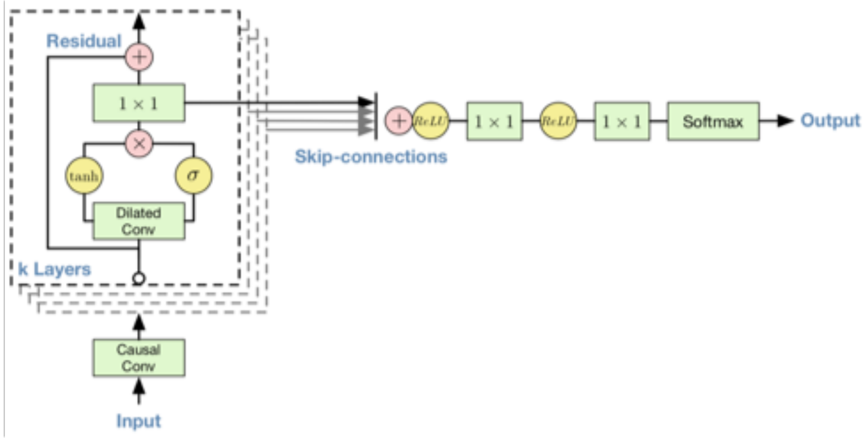


Figure from https://arxiv.org/abs/1609.03499

WaveNet

- It's common to condition WaveNet on an additional global or local input to generate
  - Speech with the characteristics of a particular speaker
  - Speech from text
  - Music

- The output conditioned on an additional input h
  - $p(x) = \prod_t p(x_t \mid x_1, \ldots, x_{t-1}, h)$

- Resulting global and local gated activation units
  - Global: $z = \tanh(W_{f,k} \circledast x + V_{f,k}^T h) \odot \sigma(W_{g,k} \circledast x + V_{g,k}^T h)$
    - Ex: A particular speaker
    - As h is fixed globally this results in a fixed offset
  - Local: $z = \tanh(W_{f,k} \circledast x + V_{f,k} \circledast h) \odot \sigma(W_{g,k} \circledast x + V_{g,k} \circledast h)$
    - Ex: linguistic features, log F_0 and phoneme durations
    - As h varies locally this changes locally
    - V is a 1x1 convolution

For audio samples see (hear?): https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Issues

- 8 bit output resolution (ideally would like 16 bit)
- Slow speed as samples are generated 1 at a time and ~ 16k are needed per second

These have been addressed in subsequent variations and extensions

Tacotron 2 ≈ Tacotron 1 + WaveNet

- Thought chain
  - Tacotron 1 is a model for generating spectrograms from text (that was used with a classical model for generating speech from spectrograms)
  - WaveNet is a model for generating audio samples that can be conditioned on different inputs
- Idea
  - Couple a modified version of Tacotron 1 spectrogram prediction with a modified version of Wavenet audio sample generation
- 2 parts to Tacotron 2
  - Part 1: recurrent sequence to sequence feature prediction network that maps characters to mel scale spectrograms
  - Part 2: modified WaveNet that maps mel scale spectrograms to audio samples

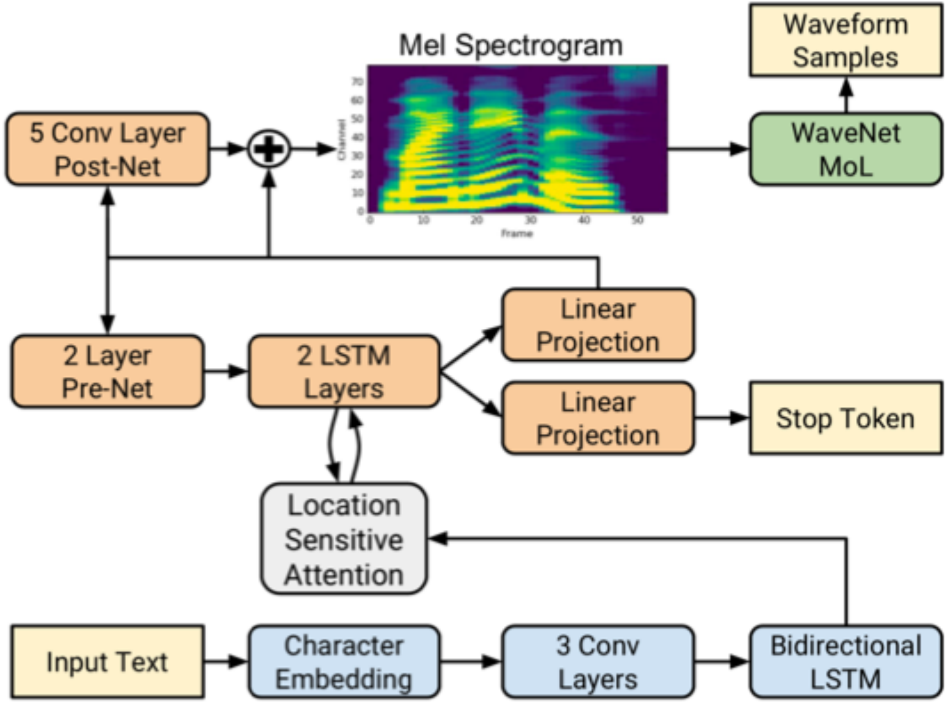


Figure from https://arxiv.org/abs/1712.05884

DeepVoice 1

DeepVoice 1 and 2 replaced components in a traditional text to speech pipeline with xNN variants

- Uses 5 neural network based components to generate speech from text
    - A grapheme to phoneme conversion model that converts from text to phonemes for text that is not in the phoneme directory
    - A segmentation model for locating phoneme boundaries based on a deep neural network with a CTC loss based on predicting the location of pairs of phonemes (thus finding their boundary)
    - A phoneme duration prediction model to predict the temporal duration of all of the phonemes
    - A fundamental frequency prediction model predicts if the phoneme is voiced and if so what is the fundamental frequency
    - An audio synthesis model based on a smaller version of WaveNet

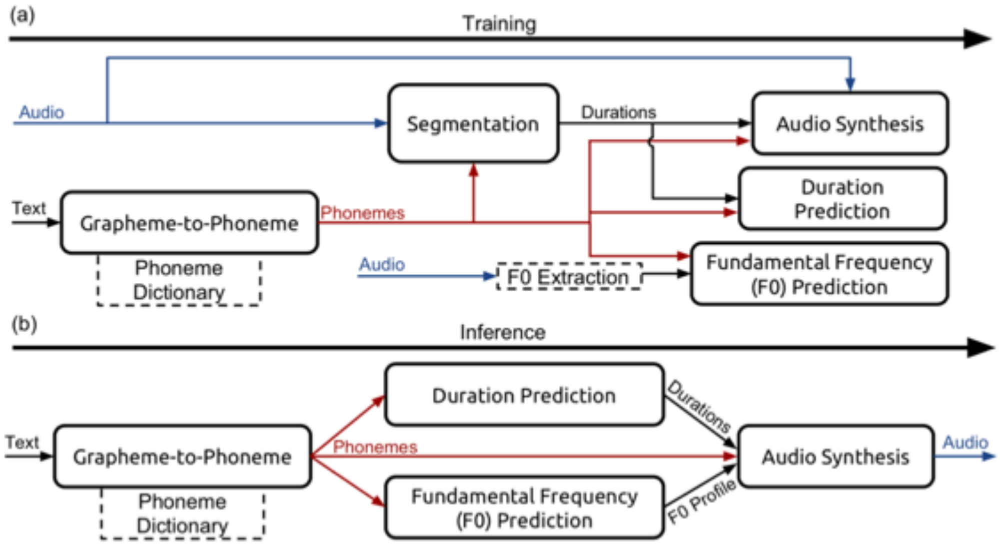


Figure from https://arxiv.org/abs/1702.07825

DeepVoice 2

Testing diagram

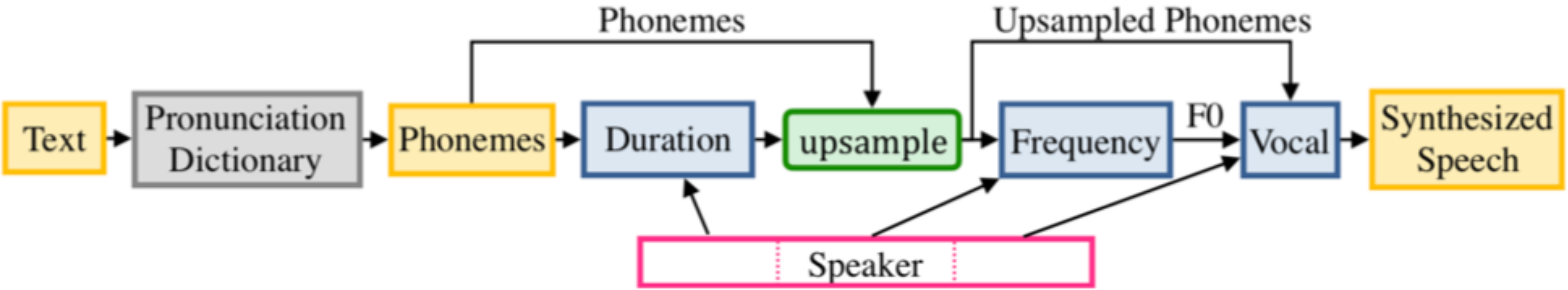


Figure from https://arxiv.org/abs/1705.08947

DeepVoice 2

Training diagram for the frequency, segmentation, duration and vocal model

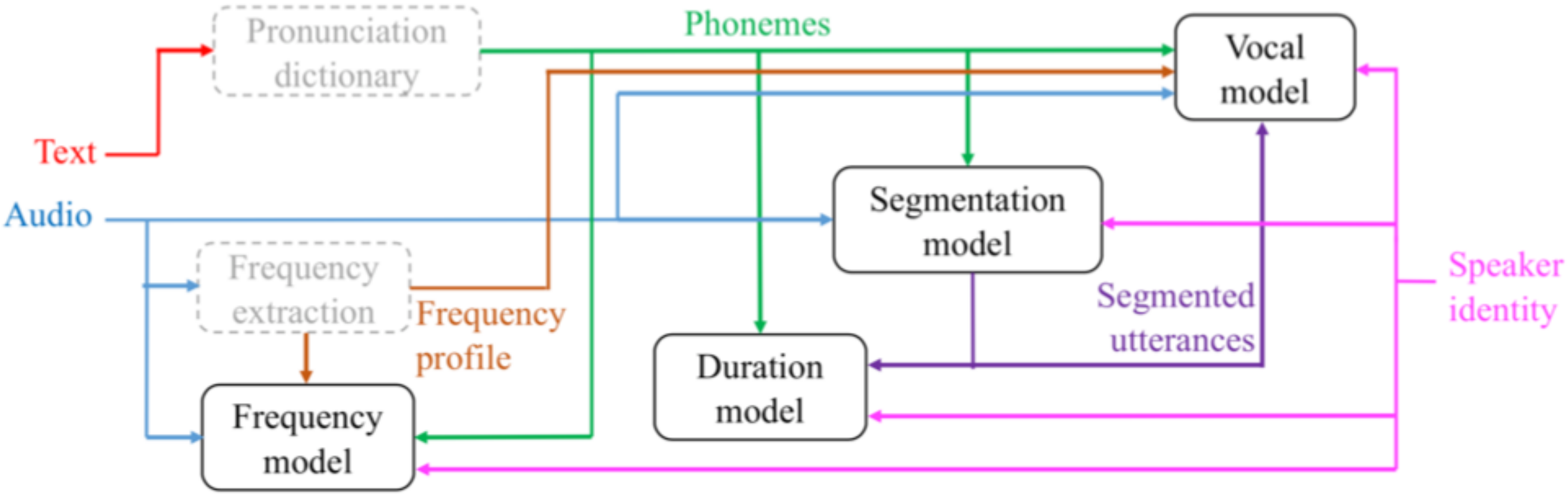


Figure from https://arxiv.org/abs/1705.08947

DeepVoice 2

Segmentation, duration and frequency model details

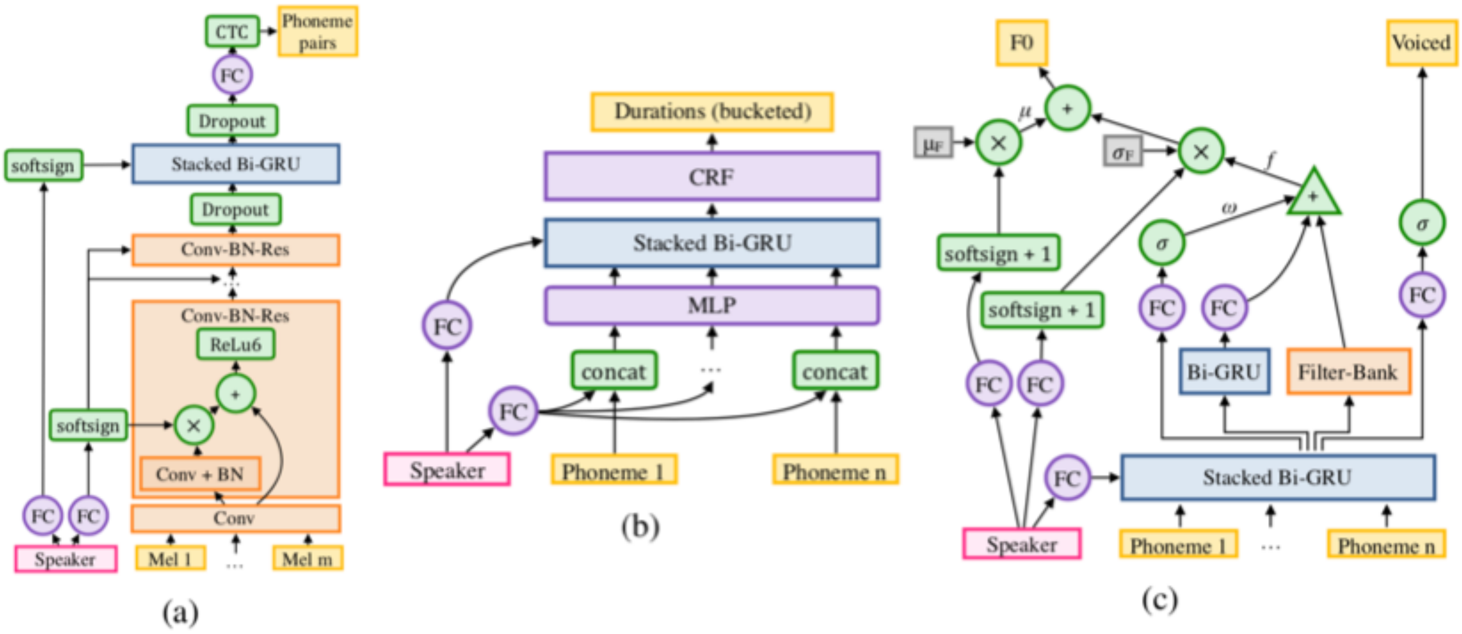


Figure 2: Architecture for the multi-speaker (a) segmentation, (b) duration, and (c) frequency model.

Figure from https://arxiv.org/abs/1705.08947

DeepVoice 3

DeepVoice 3 uses an attention based sequence to sequence architecture based on convolutional building blocks for efficiency

- Fully convolutional character to spectrogram model
  - Optimized for efficient inference on modern hardware
- Can be integrated with different audio synthesis models
  - Griffin-Lim
  - WORLD
  - WaveNet

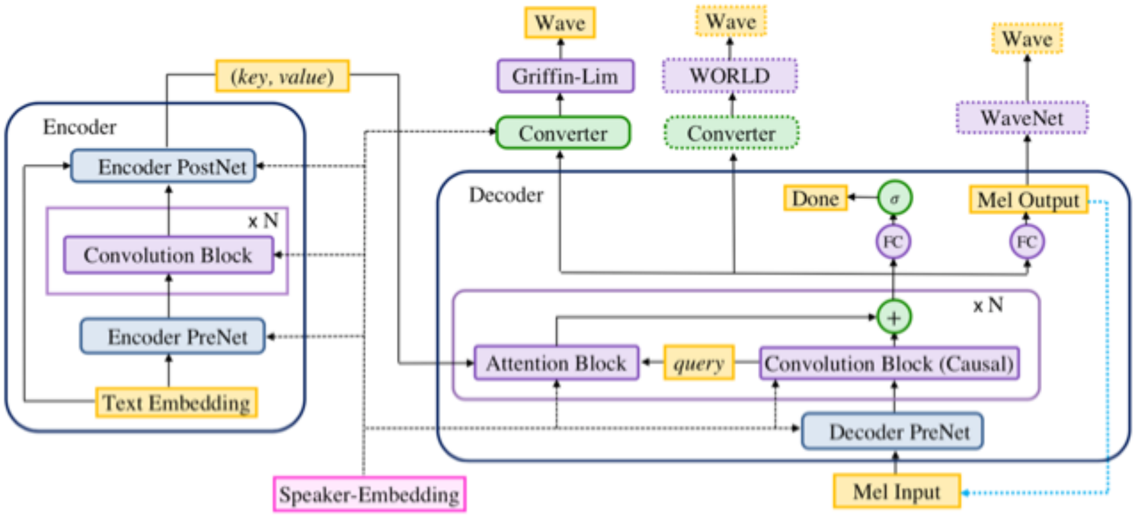


Figure from https://arxiv.org/abs/1710.07654

DeepVoice 3

DeepVoice 3 uses an attention based sequence to sequence architecture based on convolutional building blocks for efficiency

- Encoder
  - Fully convolutional encoder converts text into an internal representation
- Decoder
  - Auto regressive conversion of the internal representation with a fully convolutional causal decoder using a multi hop attention mechanism into a low dimensional audio representation (mel spectrogram)
- Converter
  - (All) vocoder parameter prediction with a fully convolutional non causal network

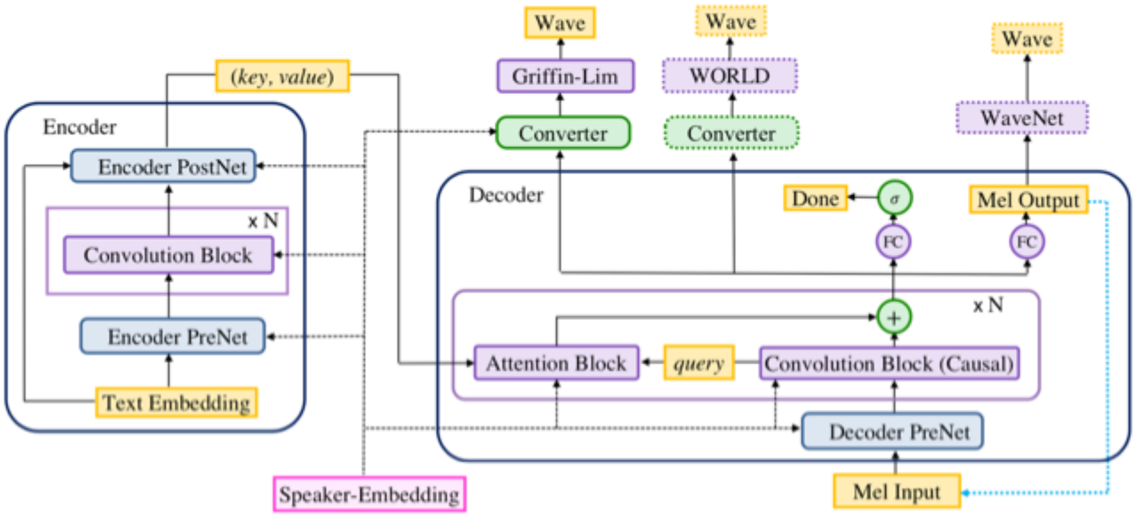
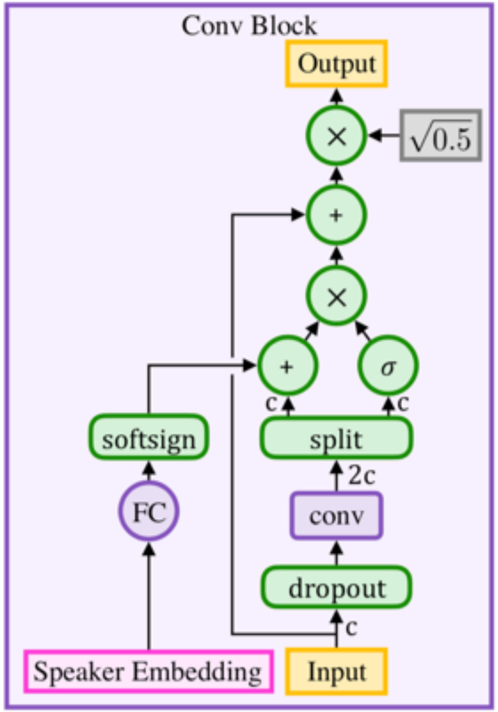


Figure from https://arxiv.org/abs/1710.07654

DeepVoice 3



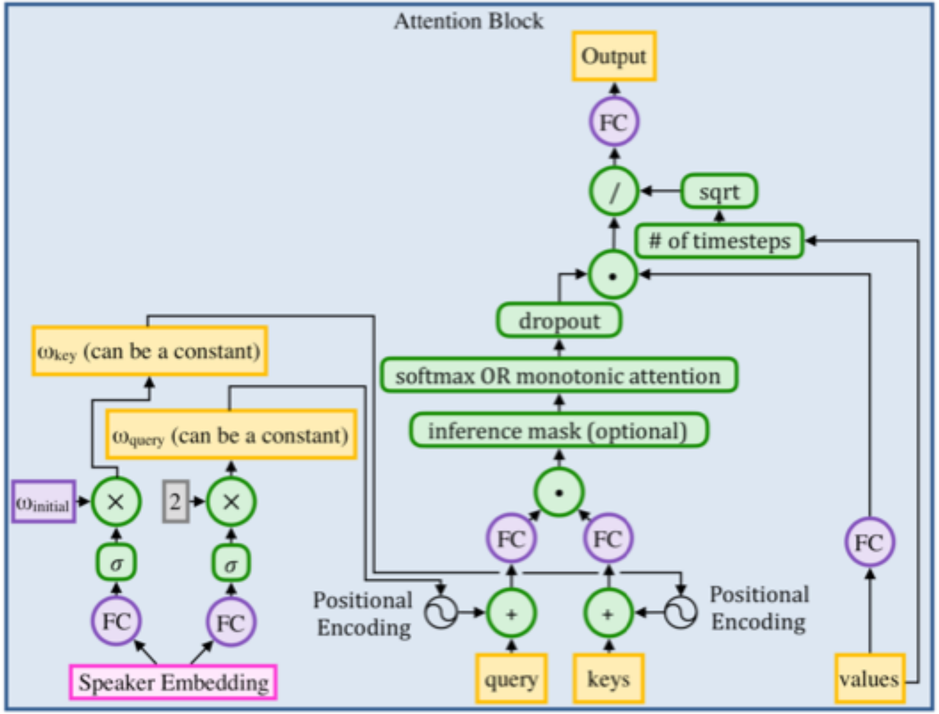
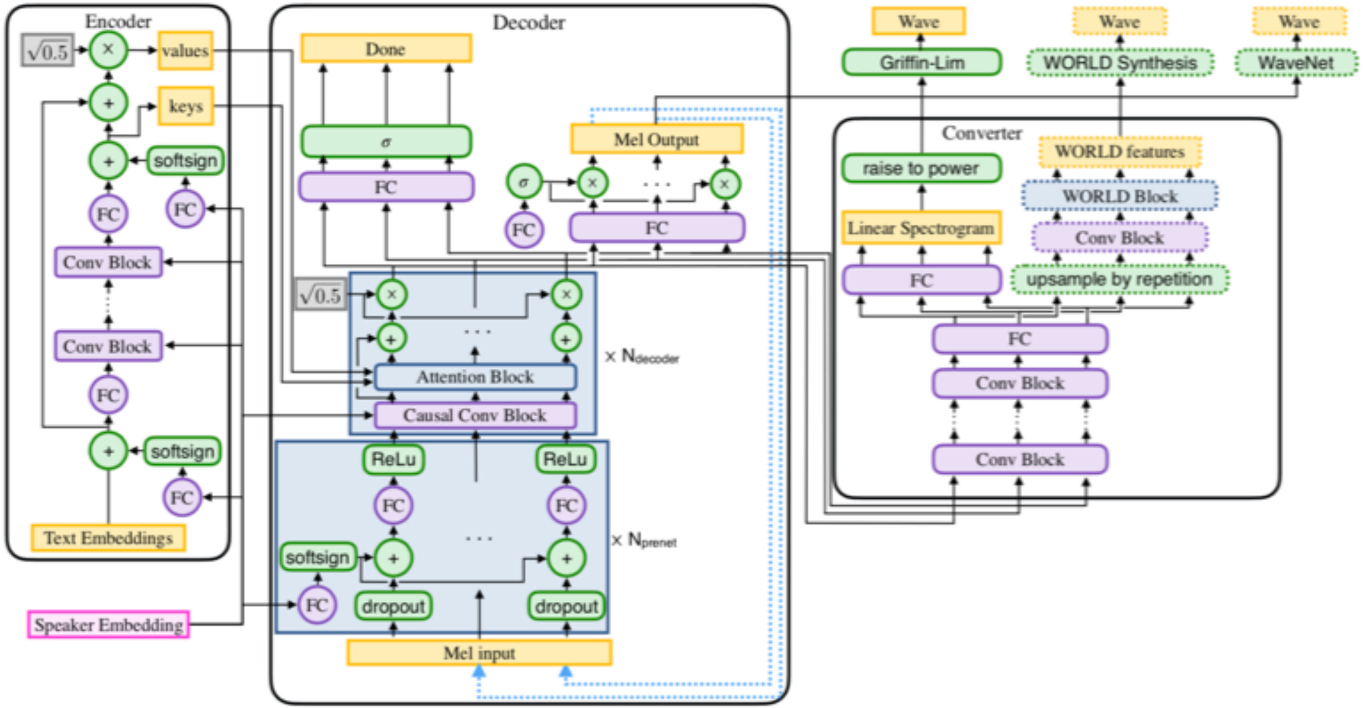


Figure from https://arxiv.org/abs/1710.07654

DeepVoice 3



Encoder
√0.5
values
keys
softsign
FC
Conv Block
Conv Block
FC
softsign
FC
Text Embeddings
Speaker Embedding
Decoder
Done
σ
FC
Mel Output
σ
FC
FC
√0.5
Attention Block
Causal Conv Block
× Ndecoder
ReLu
ReLu
FC
FC
softsign
FC
dropout
dropout
× Nprenet
Mel input
Wave
Wave
Wave
Griffin-Lim
WORLD Synthesis
WaveNet
Converter
raise to power
WORLD features
WORLD Block
Linear Spectrogram
Conv Block
FC
upsample by repetition
FC
Conv Block
Conv Block

Pointers To Some Recent Results

- This is an evolving field, a few interesting recent papers are listed below

- WaveGlow
  - WaveGlow: a flow-based generative network for speech synthesis
    - https://arxiv.org/abs/1811.00002
    - https://github.com/NVIDIA/WaveGlow
  - Glow: Generative flow with invertible 1x1 convolutions
    - https://arxiv.org/abs/1807.03039

- Voice cloning
  - Neural voice cloning with a few samples
    - https://arxiv.org/abs/1802.06006

References

Tutorials

- Stanford CS224S spoken language processing
  - http://web.stanford.edu/class/cs224s/
- Deep learning for audio
  - http://slazebni.cs.illinois.edu/spring17/lec26_audio.pdf
- Deep learning for speech recognition
  - http://lxmls.it.pt/2017/talk.pdf
- Exploring automatic speech recognition with TensorFlow
  - https://imatge.upc.edu/web/sites/default/files/pub/x.pdf
- Ten minute TensorFlow speech recognition
  - https://hackaday.com/2017/03/24/ten-minute-tensorflow-speech-recognition/
- Deep neural networks for speech processing
  - http://mi.eng.cam.ac.uk/~kmk/presentations/TutorialIC_Sep2015_part1_Knill.pdf
- Introduction to speech recognition
  - http://people.inf.ethz.ch/jaggim/meetup/3/slides/ML-Meetup-3-Dixon.pdf

Data

- VoxCeleb
  - http://www.robots.ox.ac.uk/~vgg/data/voxceleb/
- The speakers in the wild (SITW) speaker recognition database
  - https://www.sri.com/sites/default/files/publications/final2c_the_speakers_in_the_wild_28sitw29_speaker_recognition_database.pdf
- VoxForge
  - http://www.voxforge.org
- Free spoken digit dataset (FSDD)
  - https://github.com/Jakobovski/free-spoken-digit-dataset
- Speech commands: a dataset for limited-vocabulary speech recognition
  - https://arxiv.org/abs/1804.03209
- LibriSpeech ASR corpus
  - http://www.openslr.org/12/
- The design for the wall street journal-based CSR corpus
  - https://dl.acm.org/citation.cfm?id=1075614
- TED-LIUM 3: twice as much data and corpus repartition for experiments on speaker adaptation
  - https://arxiv.org/abs/1805.04699

Data

- The LJ speech dataset
    - https://keithito.com/LJ-Speech-Dataset/
- AudioSet
    - https://github.com/tensorflow/models/tree/master/research/audioset
- FMA: a dataset for music analysis
    - https://github.com/mdeff/fma
- Million song dataset
    - https://labrosa.ee.columbia.edu/millionsong/
- One billion word benchmark for measuring progress in statistical language modeling
    - https://arxiv.org/abs/1312.3005
    - http://www.statmt.org/lm-benchmark/
    - https://github.com/ciprian-chelba/1-billion-word-language-modeling-benchmark

Speech And Audio

- The speech chain
  - http://www.columbia.edu/~rmk7/HC/HC_Readings/Denes_Pinson1-2.PDF
- The 44 phonemes in English
  - https://www.dyslexia-reading-well.com/44-phonemes-in-english.html

Pre Processing

- Audio processing in TensorFlow
  - https://towardsdatascience.com/audio-processing-in-tensorflow-208f1a4103aa
- Spectrogram, cepstrum and mel-frequency analysis
  - http://www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf
- An efficient MFCC extraction method in speech recognition
  - https://ieeexplore.ieee.org/document/1692543
- Learning filterbanks from raw speech for phone recognition
  - https://arxiv.org/abs/1711.01161
- Speech processing for machine learning: filter banks, mel-frequency cepstral coefficients (MFCCs) and what's in-between
  - https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html
- SpeechPy - a library for speech processing and recognition
  - https://arxiv.org/abs/1803.01094

Network Structures

- Neural networks, types, and functional programming
  - http://colah.github.io/posts/2015-09-NN-Types-FP/
- Recurrent neural networks
  - http://www.cs.cornell.edu/courses/cs5740/2017sp/lectures/11-rnn.pdf
- NLP programming tutorial 8 - recurrent neural networks
  - http://www.phontron.com/slides/nlp-programming-en-08-rnn.pdf
- Bidirectional recurrent neural networks
  - https://pdfs.semanticscholar.org/4b80/89bc9b49f84de43acc2eb8900035f7d492b2.pdf
- Understanding LSTM networks
  - http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Essentials of deep learning: introduction to long short term memory
  - https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/
- Learning phrase representations using RNN encoder-decoder for statistical machine translation
  - https://arxiv.org/abs/1406.1078
- Capacity and trainability in recurrent neural networks
  - https://arxiv.org/abs/1611.09913

Network Structures

- Optimizing Performance of Recurrent Neural Networks on GPUs
  - https://arxiv.org/abs/1604.01946
- Hardware and software for NLP
  - https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%206%20-%20Nvidia%20RNNs%20and%20GPUs.pdf
- The fall of RNN / LSTM
  - https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0
- When recurrent models don't need to be recurrent
  - https://bair.berkeley.edu/blog/2018/08/06/recurrent/

Speaker Recognition

- Machine learning for speaker recognition
  - http://www.eie.polyu.edu.hk/~mwmak/papers/IS2016-tutorial.pdf
- Speech recognition by machine, a review
  - https://arxiv.org/abs/1001.2267
- Speaker recognition by machines and humans: a tutorial review
  - https://ieeexplore.ieee.org/document/7298570
- Improved deep speaker feature learning for text-dependent speaker recognition
  - https://arxiv.org/abs/1506.08349
- Deep speaker feature learning for text-independent speaker verification
  - https://arxiv.org/abs/1705.03670
- Text-independent speaker verification using 3d convolutional neural networks
  - https://arxiv.org/abs/1705.09422
- VoxCeleb: a large-scale speaker identification dataset
  - http://www.robots.ox.ac.uk/~vgg/publications/2017/Nagrani17/nagrani17.pdf
- VoxCeleb2: deep speaker recognition
  - https://arxiv.org/abs/1806.05622

Speaker Recognition

- Bottleneck features for speaker recognition
    - https://pdfs.semanticscholar.org/3469/fe6e53e65bced5736480afe34b6c16728408.pdf
- Improvement of distant-talking speaker identification using bottleneck features of DNN
    - https://pdfs.semanticscholar.org/b4d6/1354dc9235e26a7214cef36ab3a817b90c8a.pdf
- Analysis and optimization of bottleneck features for speaker recognition
    - http://www.odyssey2016.org/papers/pdfs_stamped/54.pdf
- A novel scheme for speaker recognition using a phonetically-aware deep neural network
    - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6853887
- Deep neural network approaches to speaker and language recognition
    - https://groups.csail.mit.edu/sls/publications/2015/Dehak_IEEE-2015.pdf
- Combining speech and speaker recognition - a joint modeling approach
    - https://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-113.pdf

Speaker Verification

- Locally-connected and convolutional neural networks for small footprint speaker recognition
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43970.pdf
- Text-independent speaker verification using 3d convolutional neural networks
  - https://arxiv.org/abs/1705.09422
- Deep neural networks for small footprint text-dependent speaker verification
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/41939.pdf
- End-to-end text-dependent speaker verification
  - https://arxiv.org/abs/1509.08062
- End-to-end attention based text-dependent speaker verification
  - https://arxiv.org/abs/1701.00562
- Deep neural network-based speaker embeddings for end-to-end speaker verification
  - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7846260

Keyword Spotting

- A whole word recurrent neural network for keyword spotting
  - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=226115
- An application of recurrent neural networks to discriminative keyword spotting
  - https://www.cs.toronto.edu/~graves/icann_santi_2007.pdf
- Discriminative keyword spotting
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/34844.pdf
- Recurrent neural networks for voice activity detection
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/41186.pdf
- Small-footprint keyword spotting using deep neural networks
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42537.pdf
- Convolutional neural networks for small-footprint keyword spotting
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43969.pdf
- Simple audio recognition
  - https://www.tensorflow.org/tutorials/sequences/audio_recognition

Keyword Spotting

- Convolutional recurrent neural networks for small-footprint keyword spotting
    - https://arxiv.org/abs/1703.05390
- Max-pooling loss training of long short-term memory networks for small-footprint keyword spotting
    - https://arxiv.org/abs/1705.02411
- Deep residual learning for small-footprint keyword spotting
    - https://arxiv.org/abs/1710.10361
- Hello edge: keyword spotting on microcontrollers
    - https://arxiv.org/abs/1711.07128
- Raw waveform-based audio classification using sample-level CNN architectures
    - https://arxiv.org/abs/1712.00866
- Speech recognition: keyword spotting through image recognition
    - https://arxiv.org/abs/1803.03759
- Efficient keyword spotting using time delay neural networks
    - https://arxiv.org/abs/1807.04353

Speech To Text Overviews

- Tensor2Tensor
  - https://github.com/tensorflow/tensor2tensor
- Speech Processing
  - http://www.speech.cs.cmu.edu/15-492/
- The application of hidden Markov models in speech recognition
  - https://mi.eng.cam.ac.uk/~mjfg/mjfg_NOW.pdf
- Deep neural networks for acoustic modeling in speech recognition
  - https://www.cs.toronto.edu/~gdahl/papers/deepSpeechReviewSPM2012.pdf
- Speech recognition
  - https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%209%20-%20Speech%20Recognition.pdf
- A comparison of sequence-to-sequence models for speech recognition
  - https://www.isca-speech.org/archive/Interspeech_2017/pdfs/0233.PDF
- Exploring neural transducers for end-to-end speech recognition
  - https://arxiv.org/abs/1707.07413

Speech To Text CTC

- Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks
  - https://www.cs.toronto.edu/~graves/icml_2006.pdf
- Towards end-to-end speech recognition with recurrent neural networks
  - http://proceedings.mlr.press/v32/graves14.pdf
- Sequence modeling with CTC
  - https://distill.pub/2017/ctc/
- Recurrent neural aligner: An encoder-decoder neural network model for sequence-to-sequence mapping
  - https://pdfs.semanticscholar.org/7703/a2c5468ecbee5b62c048339a03358ed5fe19.pdf
- Neural speech recognizer: acoustic-to-word LSTM model for large vocabulary speech recognition
  - https://arxiv.org/abs/1610.09975
- Towards end-to-end speech recognition with deep convolutional neural networks
  - https://arxiv.org/abs/1701.02720
- Exploring end-to-end techniques for low-resource speech recognition
  - https://arxiv.org/abs/1807.00868

Speech To Text Auto Segmentation

- Wav2Letter: an end-to-end convnet-based speech recognition system
  - https://arxiv.org/abs/1609.03193
- Letter-based speech recognition with gated convnets
  - https://arxiv.org/abs/1712.09444

Speech To Text RNN Transducer

- Sequence transduction with recurrent neural networks
  - https://arxiv.org/abs/1211.3711
- Speech recognition with deep recurrent neural networks
  - https://arxiv.org/abs/1303.5778
- Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer
  - https://arxiv.org/abs/1801.00841

Speech To Text Attention

- Learning phrase representations using RNN encoder-decoder for statistical machine translation
  - https://arxiv.org/abs/1406.1078
- Neural machine translation by jointly learning to align and translate
  - https://arxiv.org/abs/1409.0473
- Attention-based models for speech recognition
  - https://arxiv.org/abs/1506.07503
- Listen, attend and spell
  - https://arxiv.org/abs/1508.01211
- End-to-end attention-based large vocabulary speech recognition
  - https://arxiv.org/abs/1508.04395
- Cold fusion: training seq2seq models together with language models
  - https://arxiv.org/abs/1708.06426
- State-of-the-art speech recognition with sequence-to-sequence models
  - https://arxiv.org/abs/1712.01769
- Improved training of end-to-end attention models for speech recognition
  - https://arxiv.org/abs/1805.03294

Speech To Text Attention

- Attention and augmented recurrent neural networks
  - https://distill.pub/2016/augmented-rnns/
- An online sequence-to-sequence model using partial conditioning
  - https://papers.nips.cc/paper/6594-an-online-sequence-to-sequence-model-using-partial-conditioning.pdf

Speech To Text Other

- Analysis of CNN-based speech recognition system using raw speech as input
  - https://ronan.collobert.com/pub/matos/2015_cnnspeech_interspeech.pdf
- Speech recognition using neural networks
  - http://isl.anthropomatik.kit.edu/pdf/Tebelskis1995.pdf
- Deep convolutional neural networks for LVCSR
  - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6639347
- Improvements to deep convolutional neural networks for LVCSR
  - https://arxiv.org/abs/1309.1501
- Advances in very deep convolutional neural networks for LVCSR
  - https://arxiv.org/abs/1604.01792

Speech To Text Examples

- Deep speech: scaling up end-to-end speech recognition
  - https://arxiv.org/abs/1412.5567
  - https://github.com/mozilla/DeepSpeech
  - https://github.com/mozilla/DeepSpeech/wiki
  - https://github.com/mozilla/DeepSpeech/releases
- Deep speech 2: end-to-end speech recognition in English and Mandarin
  - https://arxiv.org/abs/1512.02595
  - https://github.com/SeanNaren/deepspeech.pytorch
- Deep speech 3
  - http://research.baidu.com/Blog/index-view?id=90

- Achieving human parity in conversational speech recognition
  - https://arxiv.org/abs/1610.05256
- The Microsoft 2017 conversational speech recognition system
  - https://arxiv.org/abs/1708.06073

Beam Search

- Beam search strategies for neural machine translation
  - https://arxiv.org/abs/1702.01806

Language Model

- A neural probabilistic language model
  - http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf
- Recurrent neural network based language model
  - http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf
- Extensions of recurrent neural network language model
  - https://pdfs.semanticscholar.org/4162/ce8bbb2fc2f0a059157f9a5b4521d6eb2af5.pdf
- A fast and simple algorithm for training neural probabilistic language models
  - https://arxiv.org/abs/1206.6426
- Towards better decoding and language model integration in sequence to sequence models
  - https://arxiv.org/abs/1612.02695

Text To Speech

- The merlin toolkit
  - http://www.cstr.ed.ac.uk/projects/merlin/
- Signal reconstruction from the STFT magnitude: a state of the art
  - http://recherche.ircam.fr/pub/dafx11/Papers/27_e.pdf

- Generative model-based text-to-speech synthesis
  - https://github.com/oxford-cs-deepnlp-2017/lectures/blob/master/Lecture%2010%20-%20Text%20to%20Speech.pdf

- Text normalization, letter to sound, prosody
  - http://web.stanford.edu/class/cs224s/lectures/224s.17.lec14.pdf
- Waveform synthesis in TTS
  - http://web.stanford.edu/class/cs224s/lectures/224s.17.lec15.pdf
- Parametric TTS, intoxication, depression, trauma, personality
  - http://web.stanford.edu/class/cs224s/lectures/224s.17.lec16.pdf

Text To Speech

- WaveNet: a generative model for raw audio
    - https://deepmind.com/blog/wavenet-generative-model-raw-audio/
    - https://arxiv.org/abs/1609.03499
- Speaker-dependent WaveNet vocoder
    - https://pdfs.semanticscholar.org/487a/a8076bf3c0edb4134759e1ddf09d64f21476.pdf
- Fast Wavenet generation algorithm
    - https://arxiv.org/abs/1611.09482
- Parallel WaveNet: fast high-fidelity speech synthesis
    - https://arxiv.org/abs/1711.10433
- WaveGlow: a flow-based generative network for speech synthesis
    - https://arxiv.org/abs/1811.00002
    - https://github.com/NVIDIA/WaveGlow
- Tacotron: towards end-to-end speech synthesis
    - https://arxiv.org/abs/1703.10135
- Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions
    - https://arxiv.org/abs/1712.05884

Text To Speech

- Deep voice: real-time neural text-to-speech
  - https://arxiv.org/abs/1702.07825
- Deep voice 2: multi-speaker neural text-to-speech
  - https://arxiv.org/abs/1705.08947
- Deep voice 3: scaling text-to-speech with convolutional sequence learning
  - https://arxiv.org/abs/1710.07654
- Deep voice 3: 2000-speaker neural text-to-speech
  - http://research.baidu.com/Blog/index-view?id=91

Text To Speech

- Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices
  - https://arxiv.org/abs/1606.06061
- Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43266.pdf
- VoiceLoop: voice fitting and synthesis via a phonolgoical loop
  - https://research.fb.com/wp-content/uploads/2018/04/voiceloop-voice-fitting-and-synthesis-via-a-phonolgoical-loop.pdf