automatic speech recognition

CSC401/2511 – Natural Language Computing – Spring 2019
Lecture 9 Frank Rudzicz
University of Toronto

Recall our input to ASR

Frame

Spectrum

Is the spectrum the best input for our ASR systems?

UNIVERSITY OF TORONTO

1. The Mel-scale filter bank

- To **mimic** the response of the **human ear** (and because it empirically improves speech recognition), we often discretize the spectrum using M triangular **filters**.
  - **Uniform** spacing before 1 kHz, **logarithmic** after 1 kHz

2. Source and filter

- The **acoustics** of speech are produced by a glottal pulse waveform (the **source**) passing through a vocal tract whose shape modifies that wave (the **filter**).

- The **shape** of the vocal tract is more important to phoneme recognition.
  - ***We to separate the source from the filter in the acoustics.***

UNIVERSITY OF TORONTO

2. Source and filter (aside)

- Since speech is assumed to be the output of a linear time invariant system, it can be described as a **convolution**.
  - **Convolution**, $x * y$, is beyond the scope of this course, but can be conceived as the modification of one signal by another.

  - For **speech signal** $x[n]$, **glottal signal** $g[n]$, and **vocal tract transfer** $v[n]$ with **spectra** $X[z]$, $G[z]$, and $V[z]$, respectively :

$$x[n] = g[n] * v[n]$$
$$X[z] = G[z]V[z]$$
$$\log X[z] = \log G[z] + \log V[z]$$

We've separated the source and filter into two terms!

2. The cepstrum

- We **separate** the **source** and the **filter** by *pretending* the log of the **spectrum** is actually a *time domain* signal.
  - the log spectrum $\log X[z]$ is a **sum** of the log spectra of the **source** and **filter**, i.e., a **superposition**;
    finding *its* spectrum will allow us to **isolate** these components.

- **Cepstrum**: *n.* the spectrum of the log of the spectrum.
  - Fun fact: 'ceps' is the reverse of 'spec'.
    Instead of '**fil**ters' we have '**lif**ters'…

2. The cepstrum

Spectrum

Log spectrum

Cepstrum

- The domain of the cepstrum is **quefrency** (a play on the word '*fre*<u>que</u>ncy').

2. The cepstrum

Spectrum

Cepstrum

Pictures from John Coleman (2005)

This is due to the vocal tract shape

This is due to the glottis

UNIVERSITY OF TORONTO

Mel-frequency cepstral coefficients

- **Mel-frequency cepstral coefficients (MFCCs)** are the most popular representation of speech used in ASR.
  - They are the **spectra** of the logarithms of the **Mel-scaled filtered spectra** of the **windows** of the **waveform**.

Advantages of MFCCs

- The cepstrum produces **highly uncorrelated features** (every dimension is useful).
  - This includes a **separation** of the **source** and **filter**.

- Historically, the cepstrum *has been* **easier to learn** than the spectrum for phoneme recognition.

- There is an efficient method to compute cepstra called the **discrete cosine transform**.

MFCCs in practice

- An observation vector of MFCCs often consists of
  - The **first 13 cepstral coefficients** (i.e., the first 13 dimensions produced by this method),
  - An additional **overall energy** measure,
  - The **velocities** (δ) of each of those 14 dimensions,
    - i.e., the rate of change of each coefficient at a given time
  - The **accelerations** ($\delta\delta$) of each of original 14 dimensions.

- The result is that at a timeframe t we have an observation MFCC vector of (13+1)*3=42 dimensions.
  - This vector is what is used by our ASR systems…

UNIVERSITY OF TORONTO

GAUSSIAN CLUSTERS

Classifying speech sounds

Note: The vowel trapezoid's dimensions were physical

- Speech sounds tend to cluster. This graph shows vowels, each in their own colour, according to the 1st two formants.

Classifying speakers

- Similarly, all of the speech produced by one **speaker** will cluster differently in **MFCC space** than speech from another speaker.
  - We can ∴ decide if a given observation comes from one speaker or another.

		Time, *t*			
		0	1	…	T
MFCC	1			…	
	2			…	
	3			…	
	…	…	…	…	…
	42			…	

Observation matrix

Fitting continuous distributions

- Since we are operating with **continuous** variables, we need to **fit continuous probability** functions to a **discrete number** of observations.

- If we *assume* the 1-dimensional data in **this histogram** is Normally distributed, we can fit a continuous Gaussian function simply in terms of the mean μ and variance σ^2.

Comparing continuous distributions

- If we observe a particular value in this univariate space, e.g., $x = 15$, we can say which of several distributions is most likely to have produced it.
  - Here, distribution B is more likely to have produced $x = 15$ because $P(x; B) > P(x; A)$.

Good fits

- Given some fixed **training data**, we want to be able to fit continuous probability functions that **best match** our observations.
  - The data in **this histogram** are **more likely** to have been produced from the parameterization on the **left**.

Univariate (1D) Gaussians

- Also known as **Normal** distributions, $N(\mu, \sigma)$

- $P(x; \mu, \sigma) = \dfrac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}\sigma}$

- The parameters we can modify are $\boldsymbol{\theta} = \langle \boldsymbol{\mu}, \boldsymbol{\sigma}^2 \rangle$
  - $\mu = E(x) = \int x \cdot P(x) dx$ **(mean)**
  - $\sigma^2 = E\left((x-\mu)^2\right) = \int (x-\mu)^2 P(x) dx$ **(variance)**

But we don't have samples for all x…

Maximum likelihood estimation

- Given data $X = \{x_1, x_2, \dots, x_n\}$, MLE produces an estimate of the parameters $\hat{\theta}$ by maximizing the **likelihood**, $L(X, \theta)$:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(X, \theta)$$

where $\boldsymbol{L}(\boldsymbol{X}, \boldsymbol{\theta}) = \boldsymbol{P}(\boldsymbol{X}; \boldsymbol{\theta}) = \prod_{i=1}^{n} P(x_i; \theta)$.

- Since $L(X, \theta)$ provides a **surface** over all $\boldsymbol{\theta}$, in order to find the **highest likelihood**, we look at the derivative

$$\frac{\delta}{\delta\theta} L(X, \theta) = 0$$

to see **at which point** the likelihood **stops growing**.

UNIVERSITY OF TORONTO

MLE with univariate Gaussians

- Estimate μ:

$$L(X,\mu) = P(X;\mu) = \prod_{i=1}^{n} P(x_i;\theta) = \prod_{i=1}^{n} \frac{\exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}\sigma}$$

$$\log L(X,\mu) = -\frac{\sum_i (x_i-\mu)^2}{2\sigma^2} - n\log\sqrt{2\pi}\sigma$$

$$\frac{\delta}{\delta\mu}\log L(X,\mu) = \frac{\sum_i (x_i-\mu)}{\sigma^2} = 0$$

$$\mu = \frac{\sum_i x_i}{n}$$

- Similarly, $\sigma^2 = \frac{\sum_i (x_i-\mu)^2}{n}$

UNIVERSITY OF TORONTO

Multivariate Gaussians

- When data is ***d*-dimensional**, the input variable is

$$\vec{x} = \langle x[1], x[2], \ldots, x[d] \rangle$$

the **mean** is

$$\vec{\mu} = E(\vec{x}) = \langle \mu[1], \mu[2], \ldots, \mu[d] \rangle$$

the **covariance matrix** is

$$\Sigma[i,j] = E(x[i]x[j]) - \mu[i]\mu[j])$$

and

$$P(\vec{x}) = \frac{\exp\left(-\frac{(\vec{x}-\vec{\mu})^{\intercal}\Sigma^{-1}(\vec{x}-\vec{\mu})}{2}\right)}{(2\pi)^{\frac{d}{2}}\,|\Sigma|^{\frac{1}{2}}}$$

$A^{\intercal}$ is the **transpose** of A
A^{-1} is the **inverse** of A
$|A|$ is the **determinant** of A

UNIVERSITY OF TORONTO

Intuitions of covariance

$\mu = [0\ 0]$
$\Sigma = \mathbf{I}$

$\mu = [0\ 0]$
$\Sigma = 0.6\mathbf{I}$

$\mu = [0\ 0]$
$\Sigma = 2.0\mathbf{I}$

- As values in Σ become larger, the Gaussian spreads out.
- ($\mathbf{I}$ is the identity matrix – 0 except for 1s on the diagonal)

Intuitions of covariance

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 0.6 \end{bmatrix}$$

- Different values on the diagonal result in different variances in their respective dimensions

UNIVERSITY OF TORONTO

Non-Gaussian observations

- Speech data is generally *not* unimodal – it's more complex.
- The observations below are **bimodal**, so fitting one Gaussian would not be representative.
  - E.g., if you usually keep your phone in your desk or on your table, it makes no sense looking for them floating in the air between them.

Mixtures of Gaussians

- **Gaussian mixture models (GMMs)** are a weighted linear combination of M component Gaussians, $\langle \Gamma_1, \Gamma_2, \ldots, \Gamma_M \rangle$:

$$P(\vec{x}) = \sum_{j=1}^{M} P(\Gamma_j) P(\vec{x} | \Gamma_j)$$

Observation likelihoods

- Assuming MFCC dimensions are independent of one another, the **covariance matrix is diagonal** – i.e., 0 off the diagonal.
- Therefore, the probability of an observation vector given a Gaussian from slide 14 becomes

$$P(\vec{x}|\Gamma_m) = \frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{d}\frac{(x[i]-\mu_m[i])^2}{\Sigma_m[i]}\right)}{(2\pi)^{\frac{d}{2}}\left(\prod_{i=1}^{d}\Sigma_m\,[i]\right)^{\frac{1}{2}}}$$

- We **imagine** a GMM first *chooses a Gaussian*, then *emits an observation* from that Gaussian.

UNIVERSITY OF TORONTO

Mixtures of Gaussians

- If we knew *which* Gaussian generated each sample, we could learn $P(\Gamma_j)$ with MLE, but that data is **hidden**, so we must use…

$$P(\vec{x}) = \sum_{j=1}^{M} P(\Gamma_j) P(\vec{x}|\Gamma_j)$$

Expectation-Maximization for GMMs

- If $\boldsymbol{\omega_m} = P(\Gamma_m)$ and $\boldsymbol{b_m}(\overrightarrow{\boldsymbol{x_t}}) = P(\overrightarrow{x_t}|\Gamma_m)$,

'component observation likelihood'

'weight'

$$P_\theta(\overrightarrow{x_t}) = \sum_{m=1}^{M} \omega_m b_m(\overrightarrow{x_t})$$

'overall probability'

where $\boldsymbol{\theta} = \langle \boldsymbol{\omega_m}, \overrightarrow{\boldsymbol{\mu_m}}, \boldsymbol{\Sigma_m} \rangle$ for $m = 1..M$

- To estimate θ, we solve $\nabla_\theta \log L(X, \theta) = 0$ where

$$\log L(X, \theta) = \sum_{t=1}^{T} \log P_\theta(\overrightarrow{x_t}) = \sum_{t=1}^{T} \log \sum_{m=1}^{M} \omega_m b_m(\overrightarrow{x_t})$$

UNIVERSITY OF TORONTO

Expectation-Maximization for GMMs

- We **differentiate** the log likelihood function w.r.t . $\mu_m[n]$ and set this to 0 to find the value of $\mu_m[n]$ at which the likelihood stops growing.

$$\frac{\delta \log L(X,\theta)}{\delta \mu_m[n]} = \sum_{t=1}^{N} \frac{1}{P_\theta(\overrightarrow{x_t})} \left[\frac{\delta}{\delta \mu_m[n]} \omega_m b_m(\overrightarrow{x_t}) \right] = 0$$

UNIVERSITY OF TORONTO

Expectation-Maximization for GMMs

- The **expectation step** gives us:

$$b_m(\overrightarrow{x_t}) = P(\overrightarrow{x_t}|\Gamma_m)$$

$$P(\Gamma_m|\overrightarrow{x_t};\theta) = \frac{\omega_m b_m(\overrightarrow{x_t})}{P_\theta(\overrightarrow{x_t})}$$

Proportion of overall probability contributed by m

- The **maximization step** gives us:

$$\widehat{\overrightarrow{\mu_m}} = \frac{\sum_t P(\Gamma_m|\overrightarrow{x_t};\theta)\overrightarrow{x_t}}{\sum_t P(\Gamma_m|\overrightarrow{x_t};\theta)}$$

$$\widehat{\Sigma_m} = \frac{\sum_t P(\Gamma_m|\overrightarrow{x_t};\theta)\overrightarrow{x_t}^2}{\sum_t P(\Gamma_m|\overrightarrow{x_t};\theta)} - \widehat{\overrightarrow{\mu_m}}^2$$

$$\widehat{\omega_m} = \frac{1}{T}\sum_{t=1}^{T} P(\Gamma_m|\overrightarrow{x_t};\theta)$$

Recall from slide 20, MLE wants:

$$\mu = \frac{\sum_i x_i}{n}$$

$$\sigma^2 = \frac{\sum_i (x_i - \mu)^2}{n}$$

UNIVERSITY OF TORONTO

Some notes…

- In the previous slide, the square of a vector, $\vec{a}^2$, is elementwise (i.e., `numpy.multiply` in Python)
  - E.g., $[2, 3, 4]^2 = [4, 9, 16]$

- Since Σ is diagonal, it can be represented as a vector.

- Can $\widehat{\overrightarrow{\sigma_m^2}} = \frac{\sum_t P\left(\Gamma_m \middle| \overrightarrow{x_t}; \theta\right) \overrightarrow{x_t}^2}{\sum_t P\left(\Gamma_m \middle| \overrightarrow{x_t}; \theta\right)} - \widehat{\overrightarrow{\mu_m}}^2$ become negative?
  - No.
    - This is left as an exercise, but only if you're interested.

Speaker recognition

- **Speaker recognition**: *n*. the identification of a speaker among several speakers given only some acoustics.

- Each **speaker** will produce speech according to **different** probability distributions.
  - We train a **Gaussian mixture model** for each speaker, given annotated data (mapping utterances to speakers).
  - We choose the speaker whose model gives the highest probability for an observation.

UNIVERSITY OF TORONTO

Recipe for GMM EM

- For each speaker, we learn a GMM given all T frames of their training data.

1. Initialize:	Guess $\theta = \langle \omega_m, \overrightarrow{\mu_m}, \Sigma_m \rangle$ for $m = 1..M$ either uniformly, randomly, or by *k*-means clustering.
2. E-step:	Compute $b_m(\overrightarrow{x_t})$ and $P(\Gamma_m \mid \overrightarrow{x_t}; \theta)$.
3. M-step:	Update parameters for $\langle \omega_m, \overrightarrow{\mu_m}, \Sigma_m \rangle$ as described on slide 30.

- (see the Reynolds & Rose (1995) paper on the course webpage for details)

SPEECH RECOGNITION

Consider what we want speech to do

...

Can we just use GMMs?

UNIVERSITY OF TORONTO

Speech databases

- Large-vocabulary continuous ASR is meant to encode full conversational speech, with a vocabulary of >64K words.
  - This requires *lots* of data to train our models.

- The **Switchboard** corpus contains 2430 conversations spread out over about 240 hours of data (~14 GB).
- The **TIMIT** database contains 63,000 sentences from 630 speakers.
  - Relatively small (~750 MB), but very popular.
- Speech data from conferences (e.g., **TED**) or from broadcast news tends to be between 3 GB and 30 GB.

UNIVERSITY OF TORONTO

Aspects of ASR systems in the world

- **Speaking mode**: **Isolated** word (e.g., "*yes*") vs. **continuous** (e.g., *"Siri, ask Cortana for the weather")*
- **Speaking style**: **Read** speech vs. **spontaneous** speech; the latter contains many **dysfluencies** (e.g., stuttering, *uh*, *like*, …)
- **Enrolment**: **Speaker-dependent** (all training data from one speaker) vs. **speaker-independent** (training data from many speakers).
- **Vocabulary**: **Small** (<20 words) or **large** (>50,000 words).
- **Transducer**: Cell phone? Noise-cancelling microphone? Teleconference microphone?

Speech is dynamic

- Speech **changes** over time.
  - GMMs are good for high-level clustering, but they encode **no notion** of *order*, *sequence*, or *time*.

- Speech is an expression of **language**.
  - We want to incorporate knowledge of how phonemes and words are ordered with **language models**.

Speech is sequences of phonemes [*]

We want to convert a series of MFCC vectors into a sequence of phonemes.

[*] not really

UNIVERSITY OF TORONTO

Phoneme dictionaries

- There are many **phonemic dictionaries** that map words to pronunciations (i.e., lists of phoneme sequences).

- The **CMU dictionary** (http://www.speech.cs.cmu.edu/cgi-bin/cmudict) is popular.
  - 127K words transcribed with the ARPAbet.
  - Includes some rudimentary **prosody markers**.

```
…
EVOLUTION          EH2 V AH0 L UW1 SH AH0 N
EVOLUTION(2)       IY2 V AH0 L UW1 SH AH0 N
EVOLUTION(3)       EH2 V OW0 L UW1 SH AH0 N
EVOLUTION(4)       IY2 V OW0 L UW1 SH AH0 N
EVOLUTIONARY       EH2 V AH0 L UW1 SH AH0 N EH2 R IY0
```

Putting it together?

Language model

Acoustic model

The noisy channel model for ASR

$$W^* = \underset{W}{\operatorname{argmax}}\, P(X|W)P(W)$$

How to encode $P(X|W)$?

Reminder – discrete HMMs

- Previously we saw **discrete HMMs**: at each state we observed a discrete symbol from a finite set of discrete symbols.

word	P(word)
ship	0.1
pass	0.05
camp	0.05
frock	0.6
soccer	0.05
mother	0.1
tops	0.05

word	P(word)
ship	0.25
pass	0.25
camp	0.05
frock	0.3
soccer	0.05
mother	0.09
tops	0.01

word	P(word)
ship	0.3
pass	0
camp	0
frock	0.2
soccer	0.05
mother	0.05
tops	0.4

Continuous HMMs (CHMM)

- A **continuous HMM** has observations that are distributed over continuous variables.
  - Observation probabilities, b_i, are also continuous.
  - E.g., here $b_0(\vec{x})$ tells us the probability of seeing the (multivariate) continuous observation $\vec{x}$ while in state 0.

Defining CHMMs

- Continuous HMMs are very similar to discrete HMMs.
  - $S = \{s_1, \dots, s_N\}$: set of states (e.g., subphones)
  - $X = \mathbb{R}^{42}$: **continuous observation space**

θ
- $\Pi = \{\pi_1, \dots, \pi_N\}$: initial state probabilities
- $A = \{a_{ij}\}, i, j \in S$: state transition probabilities
- $B = b_i(\vec{x}), i \in S, \vec{x} \in X$: **state output probabilities (i.e., Gaussian mixtures)**

yielding
- $Q = \{q_0, \dots, q_T\}, q_i \in S$: state sequence
- $\mathcal{O} = \{o_0, \dots, o_T\}, o_i \in X$: observation sequence

UNIVERSITY OF TORONTO

Word-level HMMs?

- Imagine that we want to learn an HMM for each word in our lexicon (e.g., 60K words → 60K HMMs).
- No, thank you! Zipf's law tells us that *many* words occur *very* infrequently.
  - 1 (or a few) training examples of a word is ***not*** enough to train a model as highly parameterized as a CHMM.

- In a word-level HMM, each state might be a phoneme.

Phoneme HMMs

- Phonemes *change* over time – we model these dynamics by building one HMM for *each* phoneme.
  - Tristate phoneme models are popular.
    - The centre state is often the 'steady' part.

tristate phoneme model (e.g., /oi/)

Phoneme HMMs

- We train each phoneme HMM using ***all*** sequences of that phoneme.
  - Even from different words.

```
...
64  85   ae
85  96   sh
96  102  epi
102 106  m
...
```

annotation

		Time, *t*				
		…	85	…	96	…
MFCC	1	…		…		…
	2	…		…		…
	3	…		…		…
	…	…	…	…	…	…
	42	…		…		…

observations

UNIVERSITY OF TORONTO

Combining models

- We can learn an *N*-gram **language model** from word-level transcriptions of speech data.
  - These models are discrete and are trained using MLE.

- Our phoneme HMMs together constitute our **acoustic model**.
  - Each phoneme HMM tells us how a phoneme 'sounds'.

- We can **combine** these models by **concatenating** phoneme HMMs together according to a known lexicon.
  - We use a word-to-phoneme dictionary.

Combining models

- If we know how phonemes combine to make words, we can simply **concatenate** together our phoneme models by inserting and **adjusting** transition weights.
  - e.g., *Zipf* is pronounced */z ih f/*, so…

(It's a bit more complicated than this –
normally phoneme HMMs have special 'handle' states
at either end that connect to other HMMs)

Co-articulation and triphones

- **Co-articulation**: *n.* When a phoneme is influenced by adjacent phonemes.

- A **triphone HMM** captures co-articulation.
  - Triphone model /a-b+c/ is phoneme **b** when preceded by **a** and followed by **c**.

Two (of many) triphone HMMs for /t/

/s-t+iy/ /iy-t+eh/

UNIVERSITY OF TORONTO

Combining triphone HMMs

- Triphone models can only connect to other triphone models that 'match'.

Concatenating phoneme models

We can easily incorporate unigram probabilities through transitions, too.

From Jurafsky & Martin text

UNIVERSITY OF TORONTO

Bigram models

From Jurafsky & Martin text

UNIVERSITY OF TORONTO

Using CHMMs

- As before, these HMMs are **generative** models that encode statistical knowledge of how output is **generated**.

- We **train** CHMMs with **Baum-Welch** (a type of Expectation-Maximization), as we did before with discrete HMMs.
  - Here, the observation parameters, $b_i(\vec{x})$, are adjusted using the GMM training 'recipe' from earlier.

- We find the best state sequences using **Viterbi**, as before.
  - Here, the best state sequence gives us a **sequence of phonemes** and **words**.

Audio-visual speech methods

- Observing the **vocal tract** directly, rather than through inference, can be very helpful in automatic speech recognition.

- The shape and aperture of the mouth gives some clues as to the phoneme being uttered.
  - Depending on the level of invasiveness, we can even measure the glottis and tongue directly.

Example – Lip aperture and nasals

Acoustic spectrograms

/m/

/n/

/ng/

Lip apertures over time

Coupled HMM

Where Q_i is the HMM state, m is the index into a GMM, and o_i is the observation at time i.

Nefian A V, Liang L, Pi X, *et al.* A coupled HMM for audio-visual speech recognition. In: *International Conference on Acoustics, Speech and Signal Processing ICASSP'02*. 2002. 2013–6.

NEURAL SPEECH RECOGNITION

Remember Viterbi

0
0

0.06
0

0.08
0

The best path to state s_j at time t, $\delta_j(t)$, depends on the best path to each possible previous state, $\delta_i(t-1)$, and their transitions to j, a_{ij}

$$\boldsymbol{\delta_j(t) = \max_i \left[\delta_i(t-1)a_{ij}\right] b_j(\sigma_t)}$$

$$\boldsymbol{\psi_j(t) = \operatorname{argmax}_i \left[\delta_i(t-1)a_{ij}\right]}$$

Do these probabilities need to be GMMs?

$\sigma_0 = ship$ $\sigma_1 = frock$ $\sigma_2 = tops$

Observations, σ_t

UNIVERSITY OF TORONTO

Replacing GMMs with DNNs

- Obtain $b_j(x) = \boldsymbol{p(x|q_j)}$ with a neural network.
- We can't learn that continuous distribution directly, but we can use Bayes' rule:

$$\boldsymbol{p(x|q_j)} = \frac{\boldsymbol{p(q_j|x)} \cdot p(x)}{p(q_j)}$$

Replacing GMMs with DNNs

- The probability of a word sequence W comes *loosely* from $P(X|W)$

$$\approx \max_{q_1 \cdots q_T} \prod_{t=1}^{T} P(q_t|q_{t-1})\, P(x_t|q_t) \approx \max_{q_1 \cdots q_T} \prod_{t=1}^{T} P(q_t|q_{t-1}) \frac{P(q_t|x_t)}{P(q_t)}$$

HMM

DNN

UNIVERSITY OF TORONTO

What are these DNNs learning?

- t-SNE visualizations in 2D.
- Deeper layers encode information about the **speaker**; more so given raw spectra than MFCCs (why?)

Mohamed, A., Hinton, G., & Penn, G. (2012). Understanding how deep belief networks perform acoustic modelling. In *ICASSP* (pp. 6–9).

UNIVERSITY OF TORONTO

What are these DNNs learning?

Figure 1: Multilingual BN features of five vowels from French (+), German (□) and Spanish (▽): /a/ (black), /i/ (blue), /e/ (green), /o/ (red), and /u/ (yellow)

- t-SNE visualizations of hidden layer.
- Lower layers detect **manner** of articulation

Vu, N. T., Weiner, J., & Schultz, T. (2014). Investigating the learning effect of multilingual bottle-neck features for ASR. *Interspeech*, 825–829.

UNIVERSITY OF TORONTO

Hybrid HMM and DNN

Bourlard H, Morgan. (1998) Hybrid HMM/ANN systems for speech recognition: Overview and new research directions. Adapt Process Seq Data Struct 1387:389–417. doi:10.1007/BFb0054006

Hybrid HMM and DNN

Fig. 6. *Embedded Viterbi learning with MLP.*

Results: Often, it depends on the data and task.

Sometimes, the hybrid approach is best. Sometimes, HMM-GMMs are still best, and sometimes…

Bourlard H, Morgan. (1998) Hybrid HMM/ANN systems for speech recognition: Overview and new research directions. Adapt Process Seq Data Struct 1387:389–417. doi:10.1007/BFb0054006

End-to-end neural networks

- End-to-end neural network ASR often depends on two steps:
  1. A generalization of RNNs (e.g., GRUs) to be **bi-directional.**
     This allows us to use both Forward and Backward information, as in HMMs.

Figure 2. **Bidirectional Recurrent Neural Network.**

Graves A, Jaitly N. (2014) Towards End-To-End Speech Recognition with Recurrent Neural Networks. JMLR Workshop Conf Proc, 32:1764–1772.

End-to-end neural networks

- Neural networks are typically trained at the **frame-level**.
  - This requires a separate training target for every frame, which in turn *requires the alignment between the audio and transcription sequences to be known*.
  - However, the alignment is only reliable once the classifier is trained.
- ∴, the second step for end-to-end neural network ASR is:

2. An objective function that allows sequence transcription *without* requiring prior alignment between the **input** and **target** sequences. E.g., **Connectionist Temporal Classification**:

$$CTC(x) = -\log P(y^*|x)$$ ⟸ minimize

for desired word-level transcription y^*.

Graves A, Jaitly N. (2014) Towards End-To-End Speech Recognition with Recurrent Neural Networks. JMLR Workshop Conf Proc, 32:1764–1772.

End-to-end spectra-to-characters

- **Input**: spectrograms
- **Output**: characters (incl. space and null characters)

- No phonemes or vocabulary means no OOV words.

A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, A. Ng "Deep Speech: Scaling up end-to-end speech recognition", arXiv:1412.5567v2, 2014.

End-to-end acoustic/word hybrids

- Get word boundaries from some external tool.
- Train word/characters and acoustics simultaneously.
- Obtain up to 0.11% improvement in error rates

Table 2: *Word Error Rates for the three compared models, with two different values of the beam search parameter.*

Model	WER	
	beam=11	beam=15
Baseline	10.16	9.70
Word embedding model	11.2	11.1
Combination	10.07	9.59

Bengio, S., & Heigold, G. (2014). Word Embeddings for Speech Recognition, *Interspeech*

UNIVERSITY OF TORONTO

End-to-end neural networks

Table 1. **Wall Street Journal Results.** All scores are word error rate/character error rate (where known) on the evaluation set. 'LM' is the Language model used for decoding. '14 Hr' and '81 Hr' refer to the amount of data used for training.

System	LM	14 Hr	81 Hr
RNN-CTC	None	74.2/30.9	30.1/9.2
RNN-CTC	Dictionary	69.2/30.0	24.0/8.0
RNN-CTC	Monogram	25.8	15.8
RNN-CTC	Bigram	15.5	10.4
RNN-CTC	Trigram	13.5	8.7
RNN-WER	None	74.5/31.3	27.3/8.4
RNN-WER	Dictionary	69.7/31.0	21.9/7.3
RNN-WER	Monogram	26.0	15.2
RNN-WER	Bigram	15.3	9.8
RNN-WER	Trigram	13.5	8.2
Baseline	None	—	—
Baseline	Dictionary	56.1	51.1
Baseline	Monogram	23.4	19.9
Baseline	Bigram	11.6	9.4
Baseline	Trigram	9.4	7.8
Combination	Trigram	—	6.7

DNN/HMM hybrid

Here, **lower** scores are **better**, because they are **error rates.**

But how to compute those error rates?

Graves A, Jaitly N. (2014) Towards End-To-End Speech Recognition with Recurrent Neural Networks. JMLR Workshop Conf Proc, 32:1764–1772.

EVALUATING SPEECH RECOGNITION

Evaluating ASR accuracy

- How can you tell how good an ASR system at recognizing speech?
  - E.g., if somebody said

    Reference: *how to recognize speech*

    but an ASR system heard

    Hypothesis: *how to wreck a nice beach*

    how do we quantify the error?

- One measure is **word accuracy**: *#CorrectWords/#ReferenceWords*
  - E.g., $2/4$, above
  - This runs into problems similar to those we saw with SMT.
    - E.g., the hypothesis '*how to recognize speech boing boing boing boing boing*' has 100% accuracy by this measure.
    - Normalizing by *#HypothesisWords* also has problems…

UNIVERSITY OF TORONTO

Word-error rates (WER)

- ASR enthusiasts are often concerned with **word-error rate (WER)**, which counts different **kinds** of errors that can be made by ASR at the word-level.
  - **Substitution error**: One word being mistook for another
    e.g., '***shift***' given '*ship*'
  - **Deletion error**: An input word that is 'skipped'
    e.g. '*I Torgo*' given '*I **am** Torgo*'
  - **Insertion error**: A 'hallucinated' word that was not in the input.
    e.g., '*This **Norwegian** parrot is no more*' given '*This parrot is no more*'

UNIVERSITY OF TORONTO

Evaluating ASR accuracy

- But how to decide which errors are of each type?
- E.g., Reference: *how to recognize speech*
  Hypothesis: *how to wreck a nice beach*,

- It's not so simple: '*speech*' seems to be mistaken for '*beach*', except the /s/ phoneme is incorporated into the preceding hypothesis word, '*nice*' (/n ay s/).
  - Here, '*recognize*' seems to be mistaken for '*wreck a nice*'
    - Are each of '*wreck a nice*' **substitutions** of '*recognize*'?
    - Is '*wreck*' a **substitution** for '*recognize*'?
      - If so, the words '*a*' and '*nice*' must be **insertions**.
    - Is '*nice*' a **substitution** for '*recognize*'?
      - If so, the words '*wreck*' and 'a' must be **insertions**.

UNIVERSITY OF TORONTO

Levenshtein distance

- In practice, ASR people are often more concerned with **overall** WER, and don't care about how those errors are partitioned.
  - E.g., 3 substitution errors are 'equivalent' to 1 substitution plus 2 insertions.

- The **Levenshtein** distance is a straightforward algorithm based on dynamic programming that allows us to compute overall WER.

Levenshtein distance

Allocate matrix $R[n+1, m+1]$ // where n is the number of reference words
// and m is the number of hypothesis words

Initialize $R[0,0] := 0$, **and** $R[i,j] := \infty$ **for all other** $i = 0$ **or** $j = 0$

for $i := 1..n$ // #ReferenceWords

  for $j := 1..m$ // #Hypothesis words

    $R[i,j] := \min($ $R[i-1,j]+1,$ // **deletion**

    $R[i-1,j-1],$ // if the ith reference word and
    // the jth hypothesis word **match**

    $R[i-1,j-1]+1,$ // if they differ, i.e., **substitution**

    $R[i,j-1]+1\ )$ // **insertion**

Return $100\times R[n,m]/n$

UNIVERSITY OF TORONTO

Levenshtein distance – initialization

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞						
	to	∞						
	recognize	∞						
	speech	∞						

The value at cell (i, j) is the **minimum** number of **errors** necessary to align i with j.

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0					
	to	∞						
	recognize	∞						
	speech	∞						

- $R[1,1] = \min(\infty + 1, (0), \infty + 1) = 0$ (match)
- We put a little arrow in place to indicate the choice.
  - 'Arrows' are normally stored in a **backtrace matrix**.

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞						
	recognize	∞						
	speech	∞						

- We continue along for the first reference word…
  - These are all **insertion** errors

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞	1	0	1	2	3	4
	recognize	∞						
	speech	∞						

- And onto the second reference word

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞	1	0	1	2	3	4
	recognize	∞	2	1	**1**	**2**	**3**	**4**
	speech	∞						

- Since *recognize* ≠ *wreck*, we have a **substitution** error.
- At some points, you have >1 possible path as **indicated**.
  - We can prioritize types of errors arbitrarily.

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞	1	0	1	2	3	4
	recognize	∞	2	1	1	2	3	4
	speech	∞	3	2	2	2	3	**4**

- And we finish the grid.
- There are $R[n, m] = 4$ word errors and a WER of $4/4 = 100\%$.
  - WER can be greater than 100% (relative to the reference).

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞	1	0	1	2	3	4
	recognize	∞	2	1	1	2	3	4
	speech	∞	3	2	2	2	3	**4**

- If we want, we can **backtrack** using our arrows to find the proportion of substitution, deletion, and insertion errors.

UNIVERSITY OF TORONTO

Levenshtein distance

			hypothesis					
		-	how	to	wreck	a	nice	beach
Reference	-	0	∞	∞	∞	∞	∞	∞
	how	∞	0	1	2	3	4	5
	to	∞	1	0	1	2	3	4
	recognize	∞	2	1	1	2	3	4
	speech	∞	3	2	2	2	3	**4**

- Here, we estimate 2 **substitution** errors and 2 **insertion** errors.
- Arrows can be encoded within a special **backtrace** matrix.

UNIVERSITY OF TORONTO

NEURAL SPEECH RECOGNITION (SLIGHT RETURN)

Speaker adaptation

- Given a neural ASR system trained with many speakers, we want to adapt to the voice of a new individual.
- We know how to do this with HMMs
  - (e.g., with interpolation, or (aside) with MAP or MLLR training).

- DNNs need *lots* of data to be useful, though…
  - **Conservative**: re-train whole DNN, with some constraints
  - **Transformative**: only retrain one layer (or a few)
  - **Speaker-aware**: do not really train the parameters

With notes from Hung-yi Lee

UNIVERSITY OF TORONTO

Conservative speaker adaptation

- Stopping criteria can exist on output, parameters, or meta-aspects of training

Transformative speaker adaptation

- Insert a new layer.
- Keeping all other parameters fixed, train the new ones to normalize speaker information.

- There are many alternatives…

Speaker-aware training

- Fixed length low dimension vectors, obtained in a variety of ways.
- Note we can segment things by recording device, noise, etc.
- This can be used to remove the **channel effect.**

Senior, A., & Lopez-Moreno, I. (2014). Improving DNN speaker independence with I-vector inputs. ICASSP, 225–229. https://doi.org/10.1109/ICASSP.2014.6853591

Speaker-aware training

Training data:

Speaker 1

Speaker 2

train

Acoustic features augmented with speaker vectors

Testing data:

test

All speakers use the same DNN model

Different speakers augmented by different features

UNIVERSITY OF TORONTO

Aside – the open-source Kaldi ASR

- Kaldi is the *de-facto* open-source ASR toolkit: http://kaldi-asr.org
  - It has pretrained models, including the ASpIRE chain model trained on Fisher English, augmented with impulse responses and noises to create multi-condition training.
  - My favourite incarnation uses I-Vectors to account for the speaker.
  - It often (anecdotally) performs better than Google's SpeechAPI.
  - It is originally in C++, but a wrapper (PyTorch-Kaldi) exists in the much easier Python.

Aside – convolutional neural networks

- Spectrograms are kind of images, so lets use the kinds of neural networks used in computer vision.

Next…

- We've seen how to:
  - extract useful speech features with **Mel-frequency cepstral coefficients**.
  - cluster multi-modal speech data with **Gaussian mixture models**.
  - recognize speech with **hidden Markov models** and **neural networks**.
  - evaluate ASR performance with **Levenshtein** distance.

- Next, we'll see how to **synthesize** artificial speech.

APPENDIX: CLUSTERING

(EVERYTHING THAT FOLLOWS IS AN ***ASIDE***. NOT ON THE EXAM.

(THINGS NOT ON THE EXAM MAY BE USEFUL OR AT LEAST INTERESTING))

UNIVERSITY OF TORONTO

Clustering

- **Quantization** involves turning possibly **multi-variate** and **continuous** representations into **univariate discrete** symbols.
  - Reduced storage and computation costs.
  - Potentially tremendous loss of information.

- Observation **X** is in Cluster One, so we replace it with **1**.
- Clustering is **unsupervised** learning.
  - Number and form of clusters often unknown.

Aspects of clustering

- What **defines** a particular cluster?
  - Is there some **prototype** representing each cluster?

- What defines **membership** in a cluster?
  - Usually, some distance metric $d(x, y)$ (e.g., Euclidean distance).

- How well do clusters represent **unseen data**?
  - How is a new point assigned to a cluster?
  - How do we modify that cluster as a result?

K-means clustering

- Used to group data into K clusters, $\{C_1, \dots, C_K\}$.

- Each cluster is represented by the **mean** of its assigned data.
  - (sometimes it's called the cluster's ***centroid***).

- Iterative algorithm converges to **local** optimum:
  1. **Select** K initial cluster means $\{\mu_1, \dots, \mu_K\}$ from among data points.
  2. **Until** (stopping criterion),
     a) **Assign** each data sample to closest cluster
     $$x \in C_i \quad if \quad d(x, \mu_i) \leq d(x, \mu_j), \quad \forall i \neq j$$
     b) **Update** K means from assigned samples
     $$\mu_i = E(x) \; \forall \; x \in C_i, \quad 1 \leq i \leq K$$

UNIVERSITY OF TORONTO

K-means example ($K = 3$)

- Initialize with a random selection of 3 data samples.
- Euclidean distance metric $d(x, \mu)$

K-means stopping condition

- The total **distortion**, $\mathcal{D}$, is the sum of squared error,

$$\mathcal{D} = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - \mu_i\|^2$$

- $\mathcal{D}$ decreases between n^{th} and $(n+1)^{th}$ iteration.

- We can stop training when $\mathcal{D}$ falls below some threshold $\mathcal{T}$.

$$1 - \frac{\mathcal{D}(n+1)}{\mathcal{D}(n)} < \mathcal{T}$$

UNIVERSITY OF TORONTO

Acoustic clustering example

- 12 clusters of spectra, after training.

Number of clusters

- The number of true clusters is unknown.
- We can iterate through various values of K.
  - As K approaches the size of the data, $\mathcal{D}$ approaches 0…

Hierarchical clustering

- **Hierarchical clustering** clusters data into hierarchical 'class' structures.
- Two types: top-down (**divisive**) or bottom-up (**agglomerative**).
- Often based on greedy formulations.
- Hierarchical structure can be used for hypothesizing classes.

Divisive clustering

- Creates hierarchy by successively splitting clusters into smaller groups.

Agglomerative clustering

- **Agglomerative clustering** starts with N 'seed' clusters and iteratively combines these into a hierarchy.
- On each iteration, the **two most similar** clusters are **merged** together to form a new **meta-cluster**.
- After $N - 1$ iterations, the hierarchy is complete.
- Often, when the similarity scores of new meta-clusters are tracked, the resulting graph (i.e., **dendogram**) can yield insight into the natural grouping of data.

Dendogram example

Speaker clustering

- 23 female and 53 male speakers from TIMIT.
- Data are vectors of average F1 and F2 for 9 vowels.
- Distance $d(C_i, C_j)$ is average of distances between members.

Acoustic-phonetic hierarchy

(this is basically an upside-down dendogram)

UNIVERSITY OF TORONTO

Word clustering