SPEAKER AND LANGUAGE CLASSIFICATION

SPEAKER IDENTIFICATION (SID)

SID TASK:

- 'Who said this utterance?'

SPEAKER IDENTIFICATION (SID)

SID TASK:

- 'Who said this utterance?'
- Text-dependent vs independent

SPEAKER IDENTIFICATION (SID)

SID TASK:

- 'Who said this utterance?'
- Text-dependent vs independent
  - Text-dependent: who said "Hey Google"?

SPEAKER IDENTIFICATION (SID)

SID TASK:

- 'Who said this utterance?'
- Text-dependent vs independent
  - Text-dependent: who said "Hey Google"?
  - Text-independent: who said *BLAH*?

SPEAKER IDENTIFICATION (SID)

SID TASK:

- 'Who said this utterance?'
- Text-dependent vs independent
  - Text-dependent: who said "Hey Google"?
  - Text-independent: who said *BLAH?*

More or less same toolkit used for both approaches.

CHALLENGES FOR SID

CHALLENGES FOR SID

- Need to compare utterances of varying length

CHALLENGES FOR SID

- Need to compare utterances of varying length
- Isolate speaker effects from channel effects and phonetic context

CHALLENGES FOR SID

- Need to compare utterances of varying length
- Isolate speaker effects from channel effects and phonetic context
  - Channel effect = mic quality, background noise, room acoustics, audio compression, etc.

CHALLENGES FOR SID

- Need to compare utterances of varying length
- Isolate speaker effects from channel effects and phonetic context
  - Channel effect = mic quality, background noise, room acoustics, audio compression, etc.
  - Ex: Mark or Will talking on the trolley vs. Will talking in lecture

CHALLENGES FOR SID

- Need to compare utterances of varying length
- Isolate speaker effects from channel effects and phonetic context
  - Channel effect = mic quality, background noise, room acoustics, audio compression, etc.
  - Ex: Mark or Will talking on the trolley vs. Will talking in lecture
  - Phonetic context = what words/sounds are produced

CHALLENGES FOR SID

- Need to compare utterances of varying length
- Isolate speaker effects from channel effects and phonetic context
  - Channel effect = mic quality, background noise, room acoustics, audio compression, etc.
  - Ex: Mark or Will talking on the trolley vs. Will talking in lecture
  - Phonetic context = what words/sounds are produced
  - Ex: Mark or Will saying “joint factor analysis” vs. Mark saying “let’s go to the beach”

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol
  - Be able to distinguish David and Ethan during inference

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol
  - Be able to distinguish David and Ethan during inference
  - We can do this with clustering or regression

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol
  - Be able to distinguish David and Ethan during inference
  - We can do this with clustering or regression
  - (more on this later)

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol
  - Be able to distinguish David and Ethan during inference
  - We can do this with clustering or regression
  - (more on this later)

Again, same toolkit use as for SID

SPEAKER VERIFICATION

Given speech sample X, does speech sample Y belong to the same speaker?

- Train on a closed set but infer on an **open set**
  - E.g. train on Amy, Bob and Carol
  - Be able to distinguish David and Ethan during inference
  - We can do this with clustering or regression
  - (more on this later)

Again, same toolkit use as for SID

- I generally use 'SID' as a cover term for both unless otherwise specified

TRADITIONAL PIPELINE

TRADITIONAL PIPELINE

- Audio (→ MFCCs) → 'sufficient statistics'

TRADITIONAL PIPELINE

- Audio (→ MFCCs) → 'sufficient statistics'
- Sufficient statistics → speaker modeling

TRADITIONAL PIPELINE

- Audio (→ MFCCs) → 'sufficient statistics'
- Sufficient statistics → speaker modeling
- Speaker modeling → scoring

SUFFICIENT STATISTICS

SUFFICIENT STATISTICS

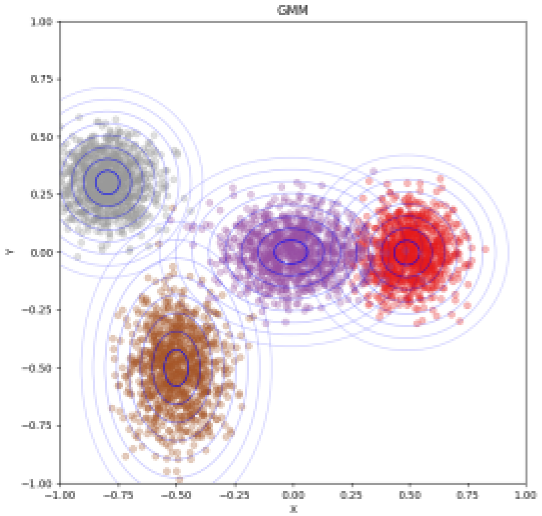
- Taken from a generative acoustic model known as the **Universal Background Model** (UBM)

SUFFICIENT STATISTICS

- Taken from a generative acoustic model known as the **Universal Background Model** (UBM)
- Typically a GMM trained on a bunch of speakers

SUFFICIENT STATISTICS

- Taken from a generative acoustic model known as the **Universal Background Model** (UBM)
- Typically a GMM trained on a bunch of speakers



SUFFICIENT STATISTICS

- Taken from a generative acoustic model known as the **Universal Background Model** (UBM)
- Typically a GMM trained on a bunch of speakers

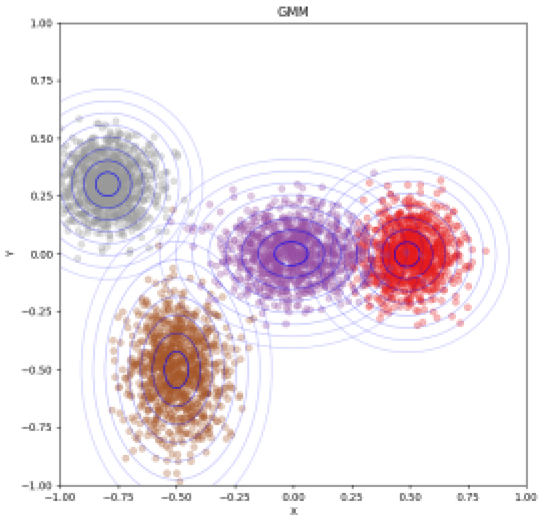


Image source: https://www.itzikbs.com/gaussian-mixture-model-gmm-3d-point-cloud-classification-primer

SPEAKER MODELING AND SCORING: OLD SCHOOL

SPEAKER MODELING AND SCORING: OLD SCHOOL

- Each speaker model also gets a GMM

SPEAKER MODELING AND SCORING: OLD SCHOOL

- Each speaker model also gets a GMM
- Score is difference between likelihood speaker GMM gives utterance and likelihood UBM gives

SPEAKER MODELING AND SCORING: OLD SCHOOL

- Each speaker model also gets a GMM
- Score is difference between likelihood speaker GMM gives utterance and likelihood UBM gives
- Define a threshold value for assigning utterance to speaker-specific GMM

SPEAKER MODELING AND SCORING: OLD SCHOOL

- Each speaker model also gets a GMM
- Score is difference between likelihood speaker GMM gives utterance and likelihood UBM gives
- Define a threshold value for assigning utterance to speaker-specific GMM

This doesn't account for channel effects!

SPEAKER MODELING WITH FA

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?
- Shrink data dimensions by just encoding these factors.

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?
- Shrink data dimensions by just encoding these factors.
    - Eigenvoice, eigenchannels

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?
- Shrink data dimensions by just encoding these factors.
  - Eigenvoice, eigenchannels
  - LDA, PCA

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?
- Shrink data dimensions by just encoding these factors.
  - Eigenvoice, eigenchannels
  - LDA, PCA
  - Other linear algebra stuff

SPEAKER MODELING WITH FA

- Pool UBM scores of whole utterance into a **supervector**
- Perform factor analysis on supervector
- What underlying factors describe this data?
- Shrink data dimensions by just encoding these factors.
  - Eigenvoice, eigenchannels
  - LDA, PCA
  - Other linear algebra stuff
  - If you’re interested in these things please come to office hours so you can explain them to me

HOW FA WORKS

If we know the channel and speaker for all utterances in our dataset, we can model the effects of both.

HOW FA WORKS

If we know the channel and speaker for all utterances in our dataset, we can model the effects of both.

- Take all recordings from a given channel (e.g. all phone conversations) and compute common factors

HOW FA WORKS

If we know the channel and speaker for all utterances in our dataset, we can model the effects of both.

- Take all recordings from a given channel (e.g. all phone conversations) and compute common factors
- Take all recordings from a given speaker and compute common factors

HOW FA WORKS

If we know the channel and speaker for all utterances in our dataset, we can model the effects of both.

- Take all recordings from a given channel (e.g. all phone conversations) and compute common factors
- Take all recordings from a given speaker and compute common factors
- In the process, learn matrices to **transform any vector** into its channel and speaker factors

TYPES OF FA

TYPES OF FA

- Joint factor analysis (JFA)

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors
- Front-End FA

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors
- Front-End FA
  - Condense UBM supervector into smaller vector of factors

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors
- Front-End FA
  - Condense UBM supervector into smaller vector of factors
  - Factor vector called 'identity vector' or *i*-vector

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors
- Front-End FA
  - Condense UBM supervector into smaller vector of factors
  - Factor vector called 'identity vector' or *i*-vector
    - For whatever reason, *i*-vector has become a common term even in architectures which don't use Front-End FA

TYPES OF FA

- Joint factor analysis (JFA)
  - Decompose UBM supervector into channel and speaker factors
- Front-End FA
  - Condense UBM supervector into smaller vector of factors
  - Factor vector called 'identity vector' or *i*-vector
    - For whatever reason, *i*-vector has become a common term even in architectures which don't use Front-End FA
  - Decompose this smaller vector into channel and speaker factors

SCORING

SCORING

- Use some kind of clustering or regression method

SCORING

- Use some kind of clustering or regression method
- Probabilistic linear discriminant analysis (PLDA)

SCORING

- Use some kind of clustering or regression method
- Probabilistic linear discriminant analysis (PLDA)
- Support vector machine (SVM)

SCORING

- Use some kind of clustering or regression method
- Probabilistic linear discriminant analysis (PLDA)
- Support vector machine (SVM)
- Spectral clustering

SCORING

- Use some kind of clustering or regression method
- Probabilistic linear discriminant analysis (PLDA)
- Support vector machine (SVM)
- Spectral clustering

Intuition: once we learn the speaker vectors, we need to learn how to put them in categories.

NNS HAVE ENTERED THE CHAT

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs
- We can also **replace the UBM and i-vector pipeline** with an ASR acoustic model

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs
- We can also **replace the UBM and i-vector pipeline** with an ASR acoustic model
  - Use bottleneck features (hidden activations from a late layer: BNFs) and phoneme probabilities

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs
- We can also **replace the UBM and i-vector pipeline** with an ASR acoustic model
  - Use bottleneck features (hidden activations from a late layer: BNFs) and phoneme probabilities
  - BNFs give rich acoustic info, phoneme probabilities account for phonetic content

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs
- We can also **replace the UBM and i-vector pipeline** with an ASR acoustic model
  - Use bottleneck features (hidden activations from a late layer: BNFs) and phoneme probabilities
  - BNFs give rich acoustic info, phoneme probabilities account for phonetic content
  - Use as input for scoring

NNS HAVE ENTERED THE CHAT

- Early applications use NNs to mimic non-neural statistical models in the same pipeline
  - E.g. replace UBM, *i*-vector model, scoring, or any combination w/ NNs
- We can also **replace the UBM and i-vector pipeline with an ASR acoustic model**
  - Use bottleneck features (hidden activations from a late layer: BNFs) and phoneme probabilities
  - BNFs give rich acoustic info, phoneme probabilities account for phonetic content
  - Use as input for scoring
  - Good news: no need to train SID, bad news: need train ASR (TANSTAAFL)!!

END-TO-END PIPELINE:

END-TO-END PIPELINE:

- Train a NN to distinguish a fixed set of speakers

END-TO-END PIPELINE:

- Train a NN to distinguish a fixed set of speakers
- Use representations from late layer as speaker vector

END-TO-END PIPELINE:

- Train a NN to distinguish a fixed set of speakers
- Use representations from late layer as speaker vector
- Basis for current SOTA SID models

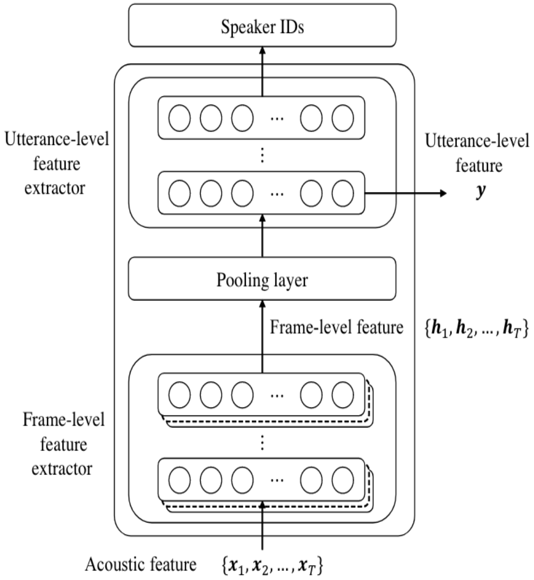


Figure 1: *DNNs for extracting utterance-level speaker features*

Okabe, Koji, Takafumi Koshinaka, and Koichi Shinoda. “Attentive Statistics Pooling for Deep Speaker Embedding.” In Interspeech 2018, 2252–56, 2018. https://doi.org/10.21437/Interspeech.2018-993.

SPEAKER DIARIZATION (SD)

Who spoke when?

SPEAKER DIARIZATION (SD)

Who spoke when?

- Predict timestamps for speaker changes

SPEAKER DIARIZATION (SD)

Who spoke when?

- Predict timestamps for speaker changes
- Predict who's speaking for each turn

CHALLENGES FOR SD

CHALLENGES FOR SD

- How to handle overlapped speech?

CHALLENGES FOR SD

- How to handle overlapped speech?
  - Multi-class labeling: predict speaker A, B, C, or any combination

CHALLENGES FOR SD

- How to handle overlapped speech?
  - Multi-class labeling: predict speaker A, B, C, or any combination
- How to handle silence/non-speech?

CHALLENGES FOR SD

- How to handle overlapped speech?
  - Multi-class labeling: predict speaker A, B, C, or any combination
- How to handle silence/non-speech?
  - Use a separate VAD model

CHALLENGES FOR SD

- How to handle overlapped speech?
  - Multi-class labeling: predict speaker A, B, C, or any combination
- How to handle silence/non-speech?
  - Use a separate VAD model
  - (During training) use 'oracle VAD' (cheating)

HOW TO DO SD

HOW TO DO SD

- Often just split audio into windows and do SID on each

HOW TO DO SD

- Often just split audio into windows and do SID on each
  - I've seen window sizes from 250ms to 3s.

HOW TO DO SD

- Often just split audio into windows and do SID on each
    - I've seen window sizes from 250ms to 3s.
- Use pipelines we just discussed with some modifications

HOW TO DO SD

- Often just split audio into windows and do SID on each
  - I've seen window sizes from 250ms to 3s.
- Use pipelines we just discussed with some modifications
- Problem: what if there's a speaker change within the window length?

END-TO-END SD

END-TO-END SD

- Train NN to cluster frames (~25ms) **within an input window** of e.g. 3-5s

END-TO-END SD

- Train NN to cluster frames (~25ms) **within an input window** of e.g. 3-5s
- Need *permutation-free training*

END-TO-END SD

- Train NN to cluster frames (~25ms) **within an input window** of e.g. 3-5s
- Need *permutation-free training*
- Given utterance $x \in [1, \ldots, t]$ model predicts label for each frame from classes A, B, C

END-TO-END SD

- Train NN to cluster frames (~25ms) **within an input window** of e.g. 3-5s
- Need *permutation-free training*
- Given utterance $x \in [1, \ldots, t]$ model predicts label for each frame from classes A, B, C
- Model rewarded so long as each speaker is distinguished from each other within the utterance

	1	**2**	**3**	**4**	**5**	**6**	**7**
Prediction	A	A	A	C	C	C	E
Label	Sally	Sally	Sally	Bob	Bob	Bob	J

	1	2	3	4	5	6	7
Prediction	A	A	A	C	C	C	B
Label	Sally	Sally	Sally	Bob	Bob	Bob	J

Correct: A=Sally, B=John, C=Bob, model is rewarded

	1	**2**	**3**	**4**	**5**	**6**	**7**
Prediction	B	B	A	C	C	C	B
Label	Vlad	Vlad	Jane	Kim	Kim	Kim	Vl

	1	2	3	4	5	6	7
Prediction	B	B	A	C	C	C	B
Label	Vlad	Vlad	Jane	Kim	Kim	Kim	Vl

Also correct: A=Jane, B=Vlad, C=Kim, model is rewarded

	1	**2**	**3**	**4**	**5**	**6**	**7**
Prediction	B	A	A	C	C	A	A
Label	Jo	Jo	Dan	Dan	Miguel	Dan	Dan

	1	2	3	4	5	6	7
Prediction	B	A	A	C	C	A	A
Label	Jo	Jo	Dan	Dan	Miguel	Dan	Dan

Incorrect: A={Jo,Dan}, B={Jo}, C={Dan,Miguel}, model fails

As long as there’s a one-to-one mapping of predicted category:label on a given utterance, the model wins

As long as there's a one-to-one mapping of predicted category:label on a given utterance, the model wins

- This is different from training the model to predict a fixed set A=Sally, B=Bob, C=John

As long as there's a one-to-one mapping of predicted category:label on a given utterance, the model wins

- This is different from training the model to predict a fixed set A=Sally, B=Bob, C=John
- If we train on enough of a variety of speakers and speaker combinations, the model will learn to diarize an open set of speakers

As long as there's a one-to-one mapping of predicted category:label on a given utterance, the model wins

- This is different from training the model to predict a fixed set A=Sally, B=Bob, C=John
- If we train on enough of a variety of speakers and speaker combinations, the model will learn to diarize an open set of speakers
- we can add in class D = silence to perform VAD alongside diarization!

After diarizing individual (e.g. 5sec) windows, cluster speakers from each window to diarize entire recording.

After diarizing individual (e.g. 5sec) windows, cluster speakers from each window to diarize entire recording.

- E.g. speaker A from window 1 sure sounds a lot like speaker C from window 5, maybe they're the same person?

SPOKEN LANGUAGE IDENTIFICATION (SLI)

SPOKEN LANGUAGE IDENTIFICATION (SLI)

- 'What language is this audio in?'

SPOKEN LANGUAGE IDENTIFICATION (SLI)

- 'What language is this audio in?'
- Also known as LID

SPOKEN LANGUAGE IDENTIFICATION (SLI)

- 'What language is this audio in?'
- Also known as LID
- I use SLI b/c LID can also be classifying language of text

CHALLENGES FOR SLI

CHALLENGES FOR SLI

- Different acoustic features useful for SLI than for SID

CHALLENGES FOR SLI

- Different acoustic features useful for SLI than for SID
- Need to distinguish language variation from speaker **and** channel variation

SLI ADVANTAGES

SLI ADVANTAGES

- Typically smaller set of distinctions to make

SLI ADVANTAGES

- Typically smaller set of distinctions to make
  - 8 billion humans on Earth

SLI ADVANTAGES

- Typically smaller set of distinctions to make
  - 8 billion humans on Earth
  - ~7k languages

SLI ADVANTAGES

- Typically smaller set of distinctions to make
  - 8 billion humans on Earth
  - ~7k languages
  - Most SLI applications will only need a small subset

SLI ADVANTAGES

- Typically smaller set of distinctions to make
  - 8 billion humans on Earth
  - ~7k languages
  - Most SLI applications will only need a small subset
- More likely that you will do inference on same subset of categories you train on

SLI ADVANTAGES

- Typically smaller set of distinctions to make
  - 8 billion humans on Earth
  - ~7k languages
  - Most SLI applications will only need a small subset
- More likely that you will do inference on same subset of categories you train on
  - E.g. if you're identifying Spanish and English sentences in bilingual movies, you just need to train on Spanish and English

LEGACY METHODS FOR SLI

LEGACY METHODS FOR SLI

- i-vector model, same as SID

LEGACY METHODS FOR SLI

- i-vector model, same as SID
- Ergodic HMM

LEGACY METHODS FOR SLI

- i-vector model, same as SID
- Ergodic HMM
  - Each state is a language

LEGACY METHODS FOR SLI

- i-vector model, same as SID
- Ergodic HMM
  - Each state is a language
  - Infer language by whichever state assigns highest probability

LEGACY METHODS FOR SLI

- i-vector model, same as SID
- Ergodic HMM
  - Each state is a language
  - Infer language by whichever state assigns highest probability
  - Doesn’t work well-a single HMM state isn’t good at modeling a whole language

- Language-specific HMM phoneme recognizers

- Language-specific HMM phoneme recognizers
  - Each state is a given phoneme from a particular language

- Language-specific HMM phoneme recognizers
  - Each state is a given phoneme from a particular language
  - Decode audio using most probable phoneme sequence to infer language

- Language-specific HMM phoneme recognizers
  - Each state is a given phoneme from a particular language
  - Decode audio using most probable phoneme sequence to infer language
  - E.g. 'perro' will be given higher probability by the phoneme sequence /pero/ for the Spanish model than /pɛɹoʊ/ from the English model

- Language-specific HMM phoneme recognizers
  - Each state is a given phoneme from a particular language
  - Decode audio using most probable phoneme sequence to infer language
  - E.g. 'perro' will be given higher probability by the phoneme sequence /pero/ for the Spanish model than /pɛɹoʊ/ from the English model
  - Works much better, but now we need phonemic labels for each language!

- Can pair w/ phonotactic LM

- Can pair w/ phonotactic LM
  - E.g. the phoneme /z/ is hardly ever followed by /l/ in English, but this is more common in Russian

- Can pair w/ phonotactic LM
  - E.g. the phoneme /z/ is hardly ever followed by /l/ in English, but this is more common in Russian
  - /zlo/ is probably not an English sequence

- Can pair w/ phonotactic LM
  - E.g. the phoneme /z/ is hardly ever followed by /l/ in English, but this is more common in Russian
  - /zlo/ is probably not an English sequence
- Phonotactic LMs are easy to train as long as we have a decent chunk of text in the language

- Can pair w/ phonotactic LM
  - E.g. the phoneme /z/ is hardly ever followed by /l/ in English, but this is more common in Russian
  - /zlo/ is probably not an English sequence
- Phonotactic LMs are easy to train as long as we have a decent chunk of text in the language
- We can even use a language-independent phone recognizer and get language ID solely from the phonotactic LMs!

END-TO-END SLI

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!
  - Good if we only need to distingush languages in training set

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!
  - Good if we only need to distingush languages in training set
- X-vector embeddings

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!
  - Good if we only need to distingush languages in training set
- X-vector embeddings
  - Same as above, trained to distinguish languages instead of speakers

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!
  - Good if we only need to distingush languages in training set
- X-vector embeddings
  - Same as above, trained to distinguish languages instead of speakers
  - Like SID, SLI x-vectors can distinguish unseen languages

END-TO-END SLI

- Direct classification: just train a NN (e.g. CNN) on an audio classification task!
  - Good if we only need to distingush languages in training set
- X-vector embeddings
  - Same as above, trained to distinguish languages instead of speakers
  - Like SID, SLI x-vectors can distinguish unseen languages
  - Use multilingual DNN BNFs as input features.

LANGUAGE DIARIZATION (LD)

"What language is spoken when?

LANGUAGE DIARIZATION (LD)

"What language is spoken when?

- E.g. codeswitching or multilingual conversation.

LANGUAGE DIARIZATION (LD)

"What language is spoken when?

- E.g. codeswitching or multilingual conversation.
- AFAIK, this has only ever been done for two-language pairs, or at least small fixed-sets. I haven’t come across any open-set LD.

LANGUAGE DIARIZATION (LD)

"What language is spoken when?

- E.g. codeswitching or multilingual conversation.
- AFAIK, this has only ever been done for two-language pairs, or at least small fixed-sets. I haven’t come across any open-set LD.
- There is way less work done on LD relative to any other task discussed today.

CHALLENGES FOR SLI

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.
- Languages *pueden cambiar* much faster *que los participantes* in a conversation

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.
- Languages *pueden cambiar* much faster *que los participantes* in a conversation
- (Languages can change much faster than participants in a conversation)

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.
- Languages *pueden cambiar* much faster *que los participantes* in a conversation
- (Languages can change much faster than participants in a conversation)
- Phonetic/phonotactic information from embedded language may be shifted towards matrix language

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.
- Languages *pueden cambiar* much faster *que los participantes* in a conversation
- (Languages can change much faster than participants in a conversation)
- Phonetic/phonotactic information from embedded language may be shifted towards matrix language
    - "fui con mi *roommate* [rumet]" (I went with my roommate)

CHALLENGES FOR SLI

- (Typically) have to distinguish *within-utterance* language changes.
- Languages *pueden cambiar* much faster *que los participantes* in a conversation
- (Languages can change much faster than participants in a conversation)
- Phonetic/phonotactic information from embedded language may be shifted towards matrix language
  - “fui con mi *roommate* [rumet]” (I went with my roommate)
  - vs. English [ɹum.mejt]

LD APPROACHES

LD APPROACHES

- Can be a side-effect of bilingual ASR

LD APPROACHES

- Can be a side-effect of bilingual ASR
    - Single ASR model, two phonesets and/or vocabularies

LD APPROACHES

- Can be a side-effect of bilingual ASR
  - Single ASR model, two phonesets and/or vocabularies
  - When decoding utterance we get LD for "free"

LD APPROACHES

- Can be a side-effect of bilingual ASR
  - Single ASR model, two phonesets and/or vocabularies
  - When decoding utterance we get LD for "free"
  - Downside: need to train bilingual ASR first! (TANSTAAFL!)

LD APPROACHES

- Can be a side-effect of bilingual ASR
  - Single ASR model, two phonesets and/or vocabularies
  - When decoding utterance we get LD for "free"
  - Downside: need to train bilingual ASR first! (TANSTAAFL!)
- Use x-vector model on short chunks

UNSUPERVISED SLI AND SID

Unsupervised representations (e.g. wav2vec2) can make SLI and SID stupidly simple

UNSUPERVISED SLI AND SID

Unsupervised representations (e.g. wav2vec2) can make SLI and SID stupidly simple

- "Hey bro, you gave me this audio to stare at for hundreds of hours, here's the language and speaker identity of each utterance" - unsupervised NN

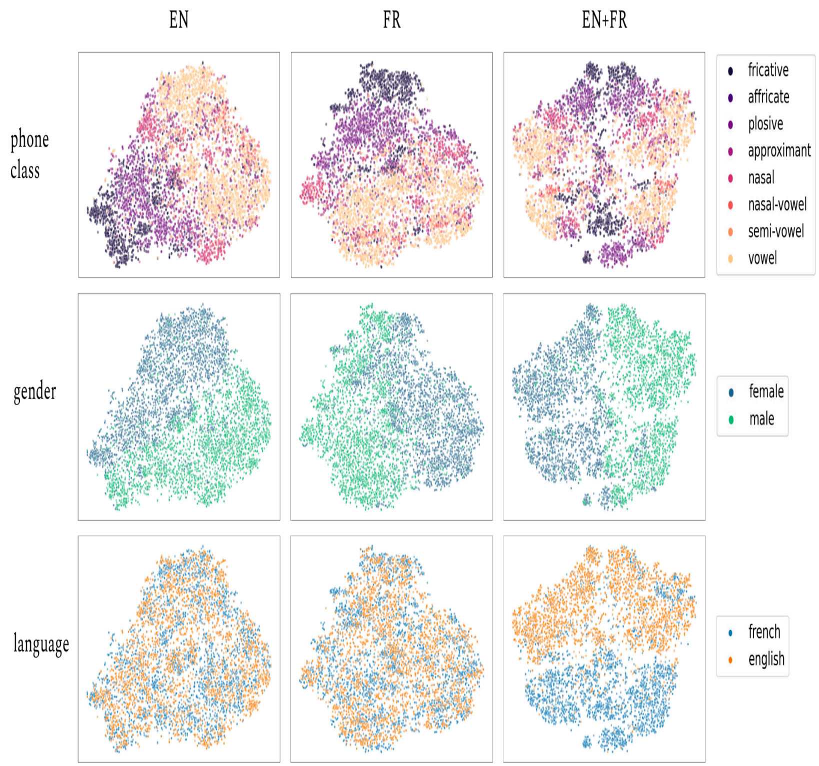


Figure 1: *T-SNE visualisation of English and French phone embeddings at the CPC level, for monolingual (EN and FR) and bilingual (EN+FR) models. Embeddings are colored based on their phone class label, gender label and language label.*

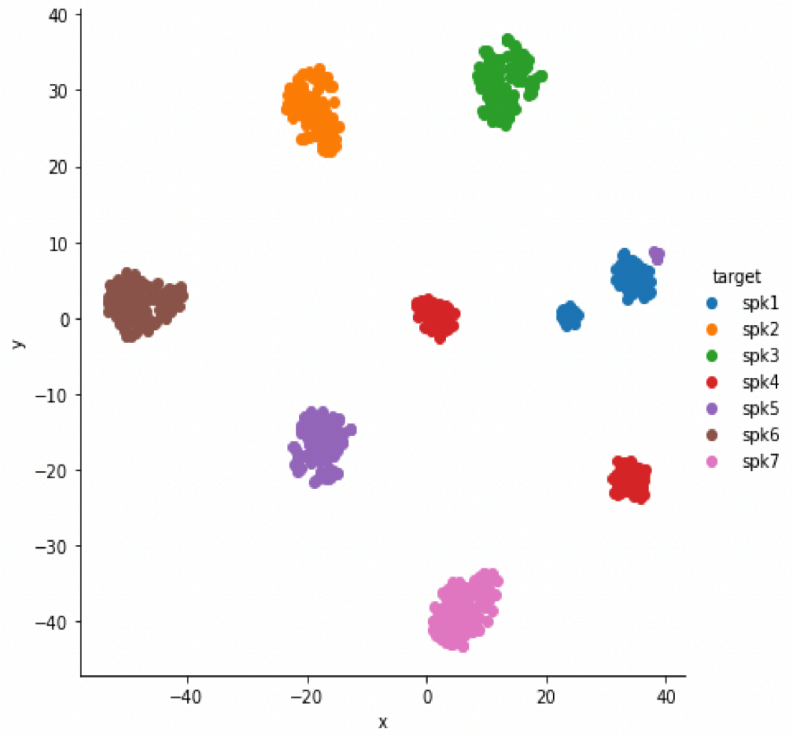


Fig. 2. t-SNE plot of the mean context vector, $\bar{c}^i$, computed for 50–10(
utterances each, from a set of 7 speakers in the LibriSpeech dataset.

WRAPPING UP

WRAPPING UP

- SID and SLI are ways of classifying whole utterances by speaker and language

WRAPPING UP

- SID and SLI are ways of classifying whole utterances by speaker and language
- SD and LD are ways of identifying speaker and language changes in continuous audio

WRAPPING UP

- SID and SLI are ways of classifying whole utterances by speaker and language
- SD and LD are ways of identifying speaker and language changes in continuous audio
- Standard method is to encode audio as some sort of vector which can be scored for its identity

WRAPPING UP

- SID and SLI are ways of classifying whole utterances by speaker and language
- SD and LD are ways of identifying speaker and language changes in continuous audio
- Standard method is to encode audio as some sort of vector which can be scored for its identity
- Neural networks allow us to engineer some fun end-to-end methods, and can also completely disrespect decades of feature engineering by performing SID and SLI *by accident.*

THANK YOU!