Speech Technology Processing Application

- Over view on Speech Synthesis
- Over view on ASR
- Accent Conversion

Speech Synthesis

Text to Speech

"Text-to-Speech software is used to convert words from a computer document (e.g. word processor document, web page) into audible speech spoken through the computer speaker"

Text to Speech Synthesis System

Text Normalization

Conversion of Text that include non standard word

- Abbreviation (like)
- Numbers (like)

Text processing

- Grapheme to Phoneme (Pronunciation)
- Prosody
  - Syllabification
  - Phrase/clause marking
  - Prosody and Intonation

Synthesis

Numbers

- Deciding how to convert numbers is another problem TTS systems have to address.
- It is a fairly simple programming challenge to convert a number into words, like 1325 becoming "one thousand three hundred twenty-five".
- However, numbers occur in many different contexts in texts, and 1325 should probably be read as "**thirteen twenty-five**" when part of an address (1325 Main St.) and as "one three two five" if it is the last four digits of a social security number.
- Often a TTS system can infer how to expand a number based on surrounding words, numbers, and punctuation, and sometimes the systems provide a way to specify the type of context if it is ambiguous.

Abbreviations

- Similarly, abbreviations like "**etc.**" are easily rendered as "et cetera", but often abbreviations can be ambiguous.
- For example, the abbreviation "**in.**" in the following example: "Yesterday it rained 3 in. Take 1 out, then put 3 in."
- "**St.**" can also be ambiguous: "St. John St."
- TTS systems with intelligent front ends can make educated guesses about how to deal with ambiguous abbreviations, while others do the same thing in all cases, resulting in nonsensical but sometimes comical outputs: "Yesterday it rained three in." or "Take one out, then put three inches."

Text-to-phoneme challenges

- Speech synthesis systems use two basic approaches to determine the pronunciation of a word based on its spelling, a process which is often called text-to-phoneme or grapheme-to-phoneme conversion, as phoneme is the term used by linguists to describe distinctive sounds in a language.

Dictionary Based approach

- The simplest approach to text-to-phoneme conversion is the **dictionary-based** approach, where a large dictionary containing all the words of a language and their correct pronunciation is stored by the program. Determining the correct pronunciation of each word is a matter of looking up each word in the dictionary and replacing the spelling with the pronunciation specified in the dictionary.

Pronunciations lexicon format of W3C (PLS)

What is Pronunciation Lexicon?

Representation of Pronunciation information of the Lexical items along with its Grapheme Representation

Why Pronunciation Lexicon ?

It is required for the development of Speech technology such as Text to Speech Synthesis and Automatic Speech Recognition

What is PLS of W3C?

The Pronunciation Lexicon Specification (PLS) is designed to enable interoperable specification of pronunciation information for both **ASR** and **TTS** engines within voice browsing applications

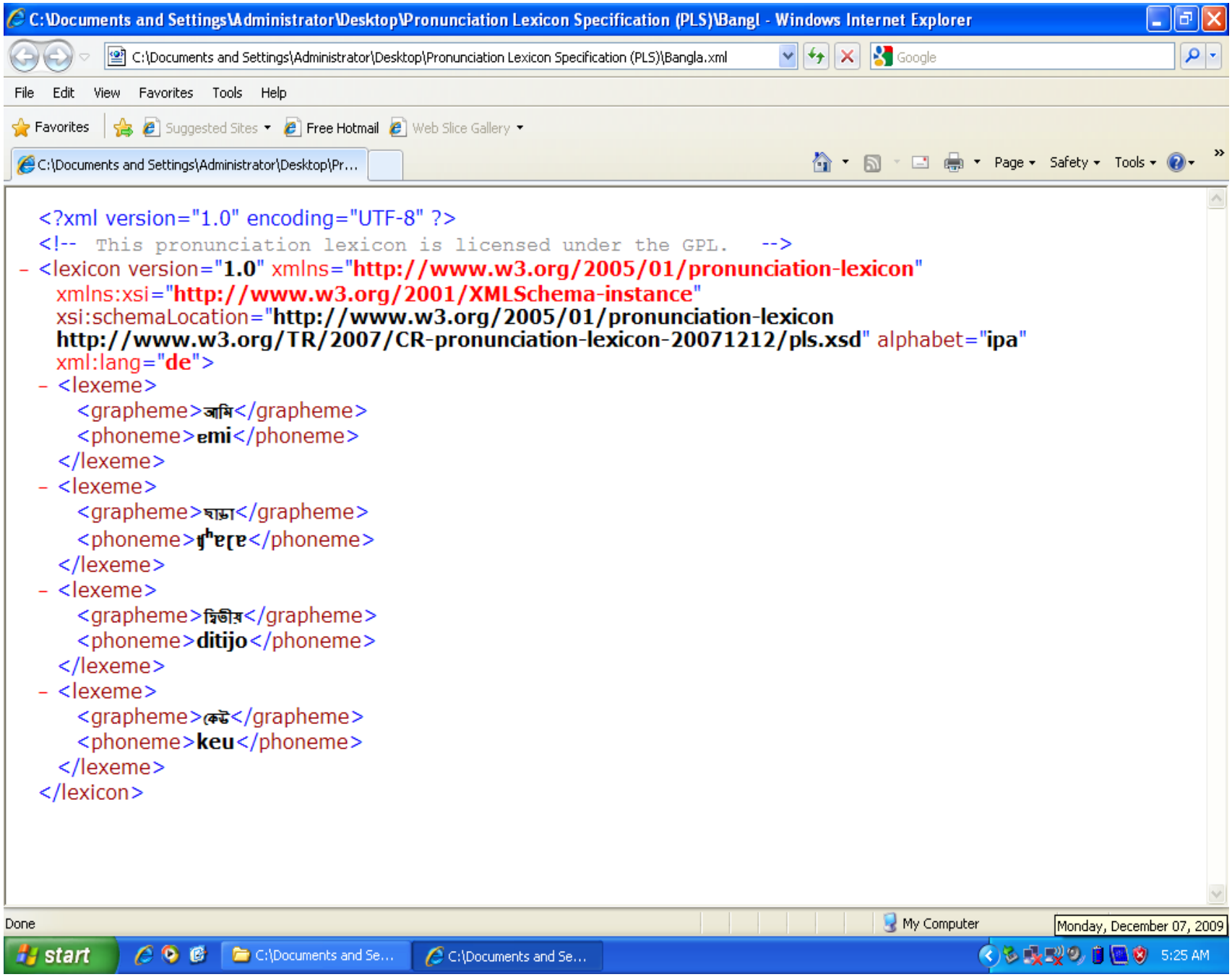
How it is used in TTS and ASR?

The PLS is the standard format of the documents referenced by the **<lexicon>** element of **SSML**. The PLS engine will load the external PLS document and transparently apply the pronunciations during the processing of the **SSML** document. An application may contain several distinct PLS documents to be used in different points of the application.

If a **pronunciation lexicon** is referenced by a **SRGS** grammar it can allow multiple pronunciations of the word in the grammar to accommodate different speaking styles

Present Pronunciation Lexicon Markup Language Definition

Elements	Attributes	Description
<lexicon>	version xml:base xmlns xml:lang alphabet	root element for PLS
<meta>	name http-equiv content	meta data container element
<metadata>		meta data container element
<lexeme>	xml:id	the container element for a single lexical entry
<grapheme>	orthography	contains orthographic information for a lexeme
<phoneme>	prefer alphabet	contains pronunciation information for a lexeme
<alias>	prefer	contains acronym expansions and words' substitutions
<example>		contains an example of the usage for a lexeme


C:\Documents and Settings\Administrator\Desktop\Pronunciation Lexicon Specification (PLS)\Bangl - Windows Internet Explorer
C:\Documents and Settings\Administrator\Desktop\Pronunciation Lexicon Specification (PLS)\Bangla.xml
File Edit View Favorites Tools Help
Favorites
Suggested Sites
Free Hotmail
Web Slice Gallery
C:\Documents and Settings\Administrator\Desktop\Pr...
Page Safety Tools
<?xml version="1.0" encoding="UTF-8" ?>
<!-- This pronunciation lexicon is licensed under the GPL. -->
<lexicon version="1.0" xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd" alphabet="ipa"
xml:lang="de">
<lexeme>
<grapheme>আমি</grapheme>
<phoneme>ɐmi</phoneme>
</lexeme>
<lexeme>
<grapheme>ঘাড়া</grapheme>
<phoneme>gʰɐɽɐ</phoneme>
</lexeme>
<lexeme>
<grapheme>দ্বিতীয়</grapheme>
<phoneme>ditijo</phoneme>
</lexeme>
<lexeme>
<grapheme>কেউ</grapheme>
<phoneme>keu</phoneme>
</lexeme>
</lexicon>
Done
My Computer
Monday, December 07, 2009
start
C:\Documents and Se...
C:\Documents and Se...
5:25 AM

Multiple pronunciations for the Same Orthography

Problem no.1→ Homographs

Homographs are words with the **Same Orthography** and **Different Meanings** and **Different Pronunciations**.

Solution under the existing PLS specification

- ❖ Using "Role" attribute under the Lexeme element
- ❖ Using "Prefer" attribute under the Phoneme element

This solution is erroneous

Proposed Solution

Proposal -I: (POS as an attribute)

The *"pos"* (parts of speech) can be an optional attribute under the phoneme element which indicates the proper pronunciation

Advantage:

- **Reduction of Lexeme numbers.**
- **Removal of Disambiguity**

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0" xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
     alphabet="ipa" xml:lang="bn">
  <lexeme>
    <grapheme>সরল</grapheme>
    <phoneme pos= “adjective”> ʃɔrol </phoneme>
    <!-- IPA string is: " ʃɔrol " -->
    <!—Itrans is: "sarala"  -->
    <!—Meaning is : "easy"  -- >
    <phoneme pos= “verb” > ʃorlo </phoneme>
    <!-- IPA string is: " ʃorlo" -->
    <!—Itrans is: "sarala"  -->
    <!—Meaning is: “moved” -- >
     <phoneme pos= “null”> ʃɔrol </phoneme>
    <!-- IPA string is: " ʃɔrol " -->
    <!—Itrans is: "sarala"  -->
    <!—Meaning is: “ easy”  -- >
  </lexeme>
</lexicon>
```

Proposal -II: (Pos as an element)

The <lexeme> element may contain optionally one or more <pos> element. Each <pos> element contains the pronunciation of the word depending on pos element information.

Advantage:

- **Reduction of Lexeme numbers.**
- **Removal of Disambiguity**

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0" xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
     alphabet="ipa" xml:lang="bn">
 <lexeme>
   <grapheme>সরল</grapheme>
   <pos> adjective </pos>
   <phoneme> ʃɔrol </phoneme>
   <!-- IPA string is: " ʃɔrol " -->
   <!—Itrans is: "sarala"  -->
   <!—Meaning is : "easy"  -- >
   <pos> verb </pos>
   <phoneme> ʃorlo </phoneme>
   <!-- IPA string is: " ʃorlo" -->
   <!—Itrans is: "sarala"  -->
   <!—Meaning is: “moved” -- >
   <pos> null </pos>
   <phoneme> ʃɔrol </phoneme>
   <!-- IPA string is: " ʃɔrol " -->
   <!—Itrans is: "sarala"  -->
   <!—Meaning is: “easy” -- >
 </lexeme>
</lexicon>
```

Problem no. 2 :

Ideolectic variation of the same orthographic information

Proposed Solution

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
xmlns=http://www.w3.org/2005/01/pronunciation-lexicon
alphabet="ipa" xml:lang="bn">
   <lexeme>
      <grapheme>ঊনত্রিশ</grapheme>
      <phoneme prefer="true">untriʃ</phoneme>
      <!-- IPA string is: "untriʃ" -->
      <phoneme>unotiriʃ</phoneme>
      <!-- IPA string is: "unotiriʃ" -->
   </lexeme>
</lexicon>
```

Multiple orthographic representation with same pronunciation

Homophones

Homophones are words with **Different Spellings** and **Different Meanings** but the **Same Pronunciation**

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      alphabet="ipa" xml:lang="bn">
  <lexeme>
    <grapheme>কুল</grapheme>
    <phoneme>kul</phoneme>
  </lexeme>
  <lexeme>
    <grapheme>কূল</grapheme>
    <phoneme>kul</phoneme>
  </lexeme>
</lexicon>
```

Need of Morphological Information

In some of the languages like Bengali, English not only **POS** information but also **Morphological** information especially in case of verb **finiteness** and **honorificity** information are very crucial in determining the pronunciation of a homograph.

For Example:

Bengali Verb “kare” has two pronunciations depending on its finiteness

Similarly Bengali verb “dhara” has two pronunciation depending upon its honorificity

English verb “read” has two pronunciations depending on its tense information

POS.s1.s2.s3.s4.s5.s6.s7.s8

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0" xmlns="http://www.w3.org/2005/01/pronunciation-
lexicon"
      alphabet="ipa" xml:lang="bn">
  <lexeme>
    <grapheme>করে</grapheme>
    <phoneme: pos= “VM.s1.s2.s3.s4.fin.s6.s7.s8 ” >kɔre</phoneme>
    <!-- IPA string is: "kɔre" -->
    <!—Itrans is: “kare” -->
    <!—Meaning is: “do/does” -- >
    <phoneme: pos= “VM.s1.s2.s3.s4.nfin.s6.s7.s8” >kore</phoneme>
    <!-- IPA string is: "kore" -->
    <!—Itrans is: “kare” -->
    <!—Meaning is: “having done” -- >
  </lexeme>
</lexicon>
```

POS→ Part of Speech Marker; s1→ person marker; s2→ tense marker; s3→ aspect marker; s4→ mood marker; s5→ finite/nonfinite; s6→ emphatic/non emphatic; s7→ negative/ non negative; s8→ honorific/ non honorific

Rule based approach

- The other approach used for text-to-phoneme conversion is the **rule-based** approach, where rules for the pronunciations of words are applied to words to work out their pronunciations based on their spellings. This is similar to the "sounding out" approach to learning reading.

Hybrid approach

- Articulatory
- Parametric
  - Formant Synthesis
  - HMM based TTS(HTS)
- Concatenative
  - Di-Phone Synthesis
  - Element Based (ESNOLA)
  - Unit selection
  - Unit Selection with Prosodic Modification
  - HMM based speech synthesis

Articulatory synthesis

- In an articulatory synthesis, models of the human articulators (tong, lips, teethes, jaw) and vocal ligament are used to simulate how an airflow passes through, to calculate what the resulting sound will be like. It is a great challenge to find good mathematical models and therefore the development of articulatory synthesis is still in research. The technique is very computation-intensive but memory requirements is almost nothing.

Formant Synthesis

This synthesis is a sort of source-filter-method that is based on mathematic models of the human speech organ. The approach pipe is modelled from a number of resonances with resemblance to the formants (frequency bands with high energy in voices) in natural speech. The first electronic voices Voder, and later on OVE and PAT, were speaking with totally synthetic and electronic produced sounds using formant synthesis. As with articulatory synthesis, the memory consumption is small but CPU usage is large.

Formant Synthesis

- Formant synthesis does not use any human speech samples at runtime. Instead, the output synthesized speech is created using an acoustic model.
- Parameters such as frequency amplitude etc are varied over time to create a waveform of artificial speech.

Concatenating synthesis

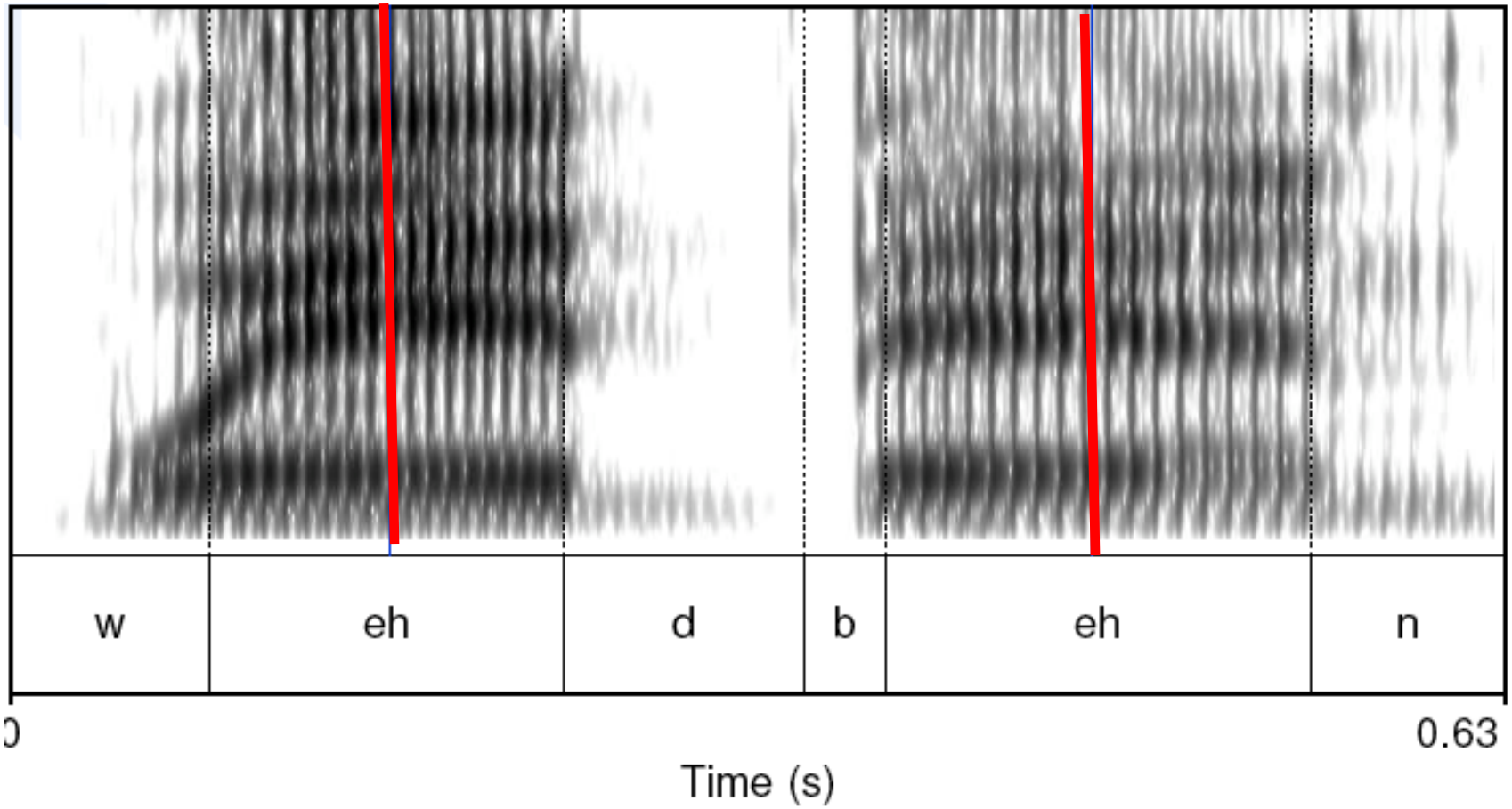
- A concatenating synthesis is made of recorded pieces of speech (sound-clips) that is then unitized and formed to speech.
- Depending on the length of sound-clips that are used it become a diphone or a polyphonic synthesis.
- More developed version is also called a Unit Selection synthesis, where the synthesizer has access to both long and short segments of speech and the best segments for the actual context is chosen

Diphone TTS architecture

- **Training:**
  - Choose units (kinds of diphones)
  - Record 1 speaker saying 1 example of each diphone
  - Mark the boundaries of each diphones,
    - cut each diphone out and create a diphone database
- **Synthesizing an utterance,**
  - grab relevant sequence of diphones from database
  - Concatenate the diphones, doing slight signal processing at boundaries
  - use signal processing to change the prosody (F0, energy, duration) of selected sequence of diphones

Diphones

Mid-phone is more stable than edge:



Designing a diphone inventory: Nonsense words

- **Build set of carrier words:**
  - pau t aa b aa b aa pau
  - pau t aa m aa m aa pau
  - pau t aa m iy m aa pau
  - pau t aa m iy m aa pau
  - pau t aa m ih m aa pau
- **Advantages:**
  - Easy to get all diphones
  - Likely to be pronounced consistently
    - No lexical interference
- **Disadvantages:**
  - (possibly) bigger database
  - Speaker becomes bored

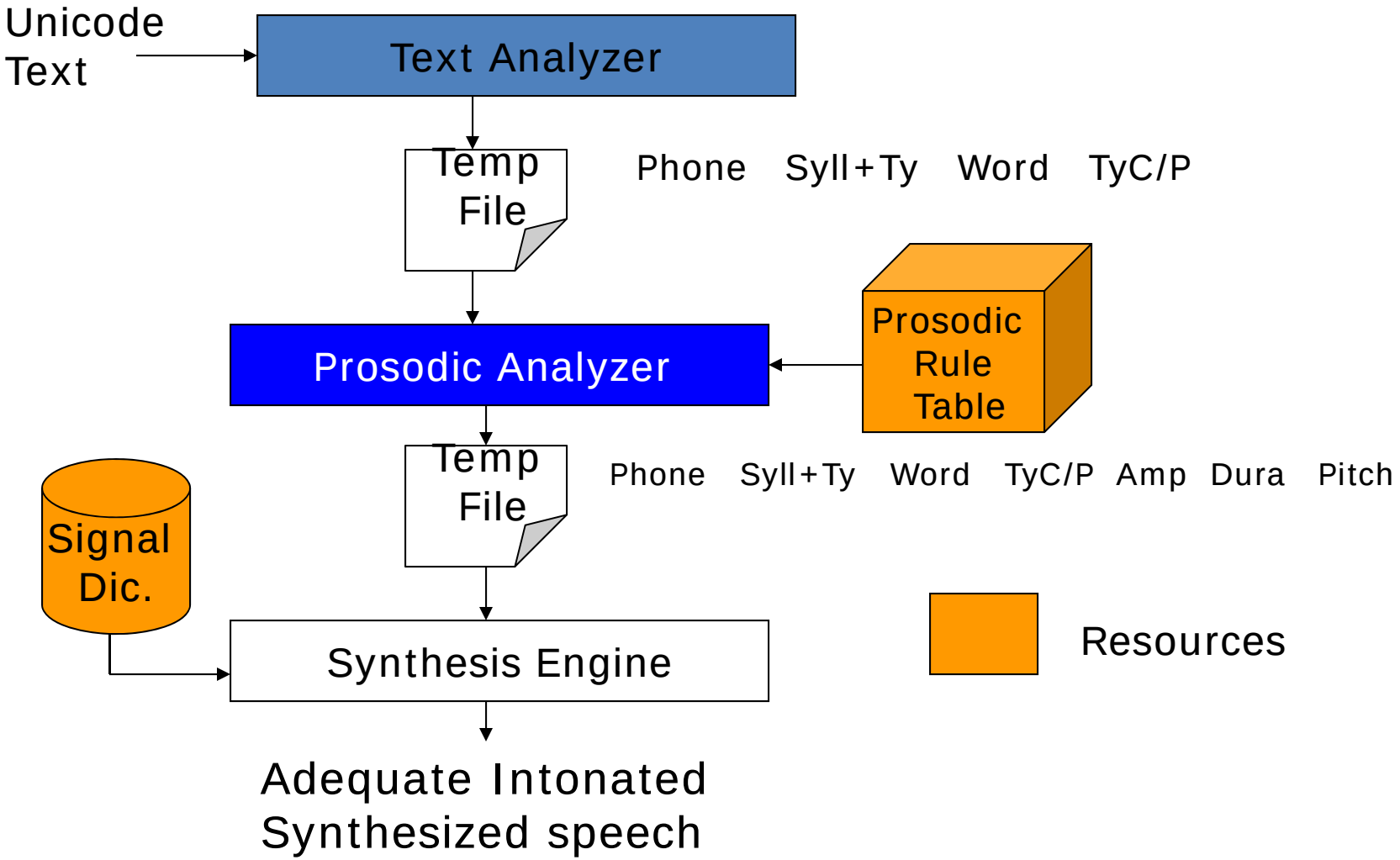
Designing a diphone inventory: Natural words

- Greedily select sentences/words:
  - Quebecois arguments
  - Brouhaha abstractions
  - Arkansas arranging
- Advantages:
  - Will be pronounced naturally
  - Easier for speaker to pronounce
  - Smaller database? (505 pairs vs. 1345 words)
- Disadvantages:
  - May not be pronounced correctly

Making recordings consistent:

- Diiphone should come from mid-word
  - Help ensure full articulation
- Performed consistently
  - Constant pitch (monotone), power, duration
- Use (synthesized) prompts:
  - Helps avoid pronunciation problems
  - Keeps speaker consistent
  - Used for alignment in labeling

Architecture of the ESNOLA System



1. CVCV. ⟶ C +CV+V+VC+C+V+V_o

बाजे */baaje/* ⟶ b+baa +aa+aaj+j+je+e+e_o

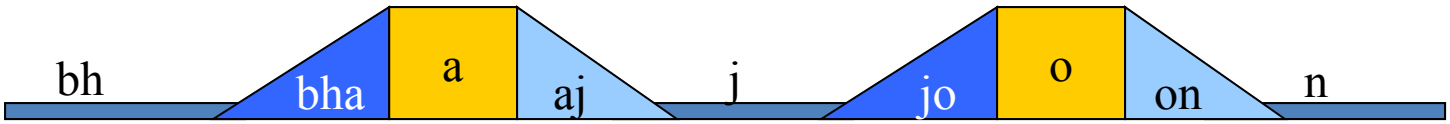
2. VCV. V_i+V +VC+C+CV+V+V_o

आगे */aage/* ⟶ aa_i+aa+aag+g+ge+e+e_o

3. CVYV. C +CV+V+VY+YV+V_o

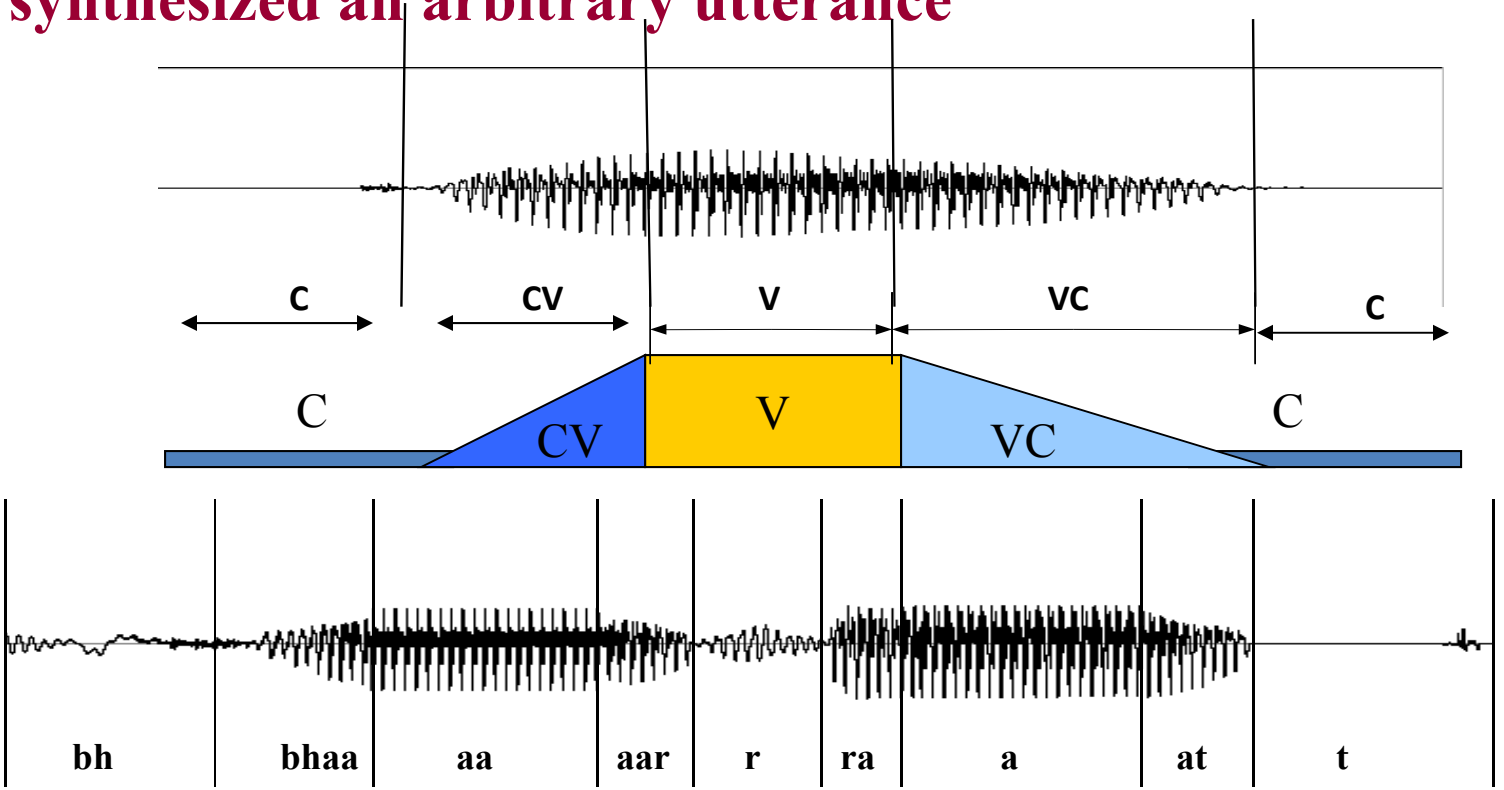
रोयो /royo/ ⟶ r+ro+o+oy+y+yo+o+o_o

भजन् */bhajon/*



ESNOLA Method Based Speech Synthesis System for Bangla

Concatenative synthesis is based on putting together pieces (acoustic unit) of natural(recorded) speech to synthesized an arbitrary utterance



/bhaarat/

* **Epoch Synchronous Non-OverLapping Add**

রাজা মহানন্দ রাজধানীতে তৈরি করেছিল শিব মন্দির ও বৈষ্ণবদের মন্দির।

Unit Selection Synthesis

- Generalization of the diphone intuition
  - Larger units
    - From diphones to sentences
  - Many copies of each unit
    - 10 hours of speech instead of 1500 diphones (a few minutes of speech)
  - No signal processing applied to each unit
    - Unlike diphones

Why Unit Selection Synthesis

- Natural data solves problems with diphones
  - Diphone databases are carefully designed but:
    - Speaker makes errors
    - Speaker doesn't speak intended dialect
    - Require database design to be right
  - If it's automatic
    - Labeled with what the speaker actually said
    - Coarticulation, schwas, flaps are natural
- "There's no data like more data"
  - Lots of copies of each unit mean you can choose just the right one for the context
  - Larger units mean you can capture wider effects

Unit Selection Intuition

- Given a big database
- For each segment (diphone) that we want to synthesize
  - Find the unit in the database that is the *best* to synthesize this target segment
- What does "best" mean?
  - "Target cost": Closest match to the target description, in terms of
    - Phonetic context
    - F0, stress, phrase position
  - "Join cost": Best join with neighboring units
    - Matching formants + other spectral characteristics
    - Matching energy
    - Matching F0

$$C(t_1^n, u_1^n) = \sum_{i=1}^{n} C^{target}(t_i, u_i) + \sum_{i=2}^{n} C^{join}(u_{i-1}, u_i)$$

Targets and Target Costs

- A measure of how well a particular unit in the database matches the internal representation produced by the prior stages
- Features, costs, and weights
- Examples:
  - /ih-t/ from stressed syllable, phrase internal, high F0, content word
  - /n-t/ from unstressed syllable, phrase final, low F0, content word
  - /dh-ax/ from unstressed syllable, phrase initial, high F0, from function word “the”

Target Costs

- Comprised of k subcosts
  - Stress
  - Phrase position
  - F0
  - Phone duration
  - Lexical identity
- Target cost for a unit:

$$C^t(t_i, u_i) = \sum_{k=1}^{p} w_k^t C_k^t(t_i, u_i)$$

How to set target cost weights (1)

- What you REALLY want as a target cost is the perceivable acoustic difference between two units
- But we can't use this, since the target is NOT ACOUSTIC yet, we haven't synthesized it!
- We have to use features that we get from the TTS upper levels (phones, prosody)
- But we DO have lots of acoustic units in the database.
- We could use the acoustic distance between these to help set the WEIGHTS on the acoustic features.

How to set target cost weights (2)

- Clever Hunt and Black (1996) idea:
- Hold out some utterances from the database
- Now synthesize one of these utterances
  - Compute all the phonetic, prosodic, duration features
  - Now for a given unit in the output
  - For each possible unit that we COULD have used in its place
  - We can compute its acoustic distance from the TRUE ACTUAL HUMAN utterance.
  - This acoustic distance can tell us how to weight the phonetic/prosodic/duration features

How to set target cost weights (3)

- Hunt and Black (1996)
- Database and target units labeled with:
  - phone context, prosodic context, etc.
- Need an acoustic similarity between units too
- Acoustic similarity based on perceptual features
  - MFCC (spectral features)
  - F0 (normalized)
  - Duration penalty

$$AC^t(t_i, u_i) = \sum_{i=1}^{p} w_i^a abs(P_i(u_n) - P_i(u_m)$$

How to set target cost weights (3)

- Collect phones in classes of acceptable size
  - E.g., stops, nasals, vowel classes, etc
- Find AC between all of same phone type
- Find C^t between all of same phone type
- Estimate $w_{1\text{-}j}$ using linear regression

How to set target cost weights (4)

- Target distance is
$$C^t(t_i,u_i) = \sum_{k=1}^{p} w_k^t C_k^t(t_i,u_i)$$
- For examples in the database, we can measure
$$AC^t(t_i,u_i) = \sum_{i=1}^{p} w_i^a abs(P_i(u_n) - P_i(u_m)$$
- Therefore, estimate weights w from all examples of
$$AC^t(t_i,u_i) \approx \sum_{k=1}^{p} w_k^t C_k^t(t_i,u_i)$$
- Use linear regression

Richard Sproat slide

Join (Concatenation) Cost

- Measure of smoothness of join
- Measured between two database units (target is irrelevant)
- Features, costs, and weights
- Comprised of k subcosts:
  - Spectral features
  - F0
  - Energy
- Join cost:

$$C^j(u_{i-1}, u_i) = \sum_{k=1}^{p} w_k^j C_k^j(u_{i-1}, u_i)$$

Slide from Paul Taylor

Unit selection

- The greatest difference between a Unit selection and a diphone voice is the length of the used speech segments.
- There are entire words and phrases stored in the unit database. this implies that the database for the Unit selection voices are many times bigger than for diphone voices.
- Thus, the memory consumption is huge while the CPU consumption is low.

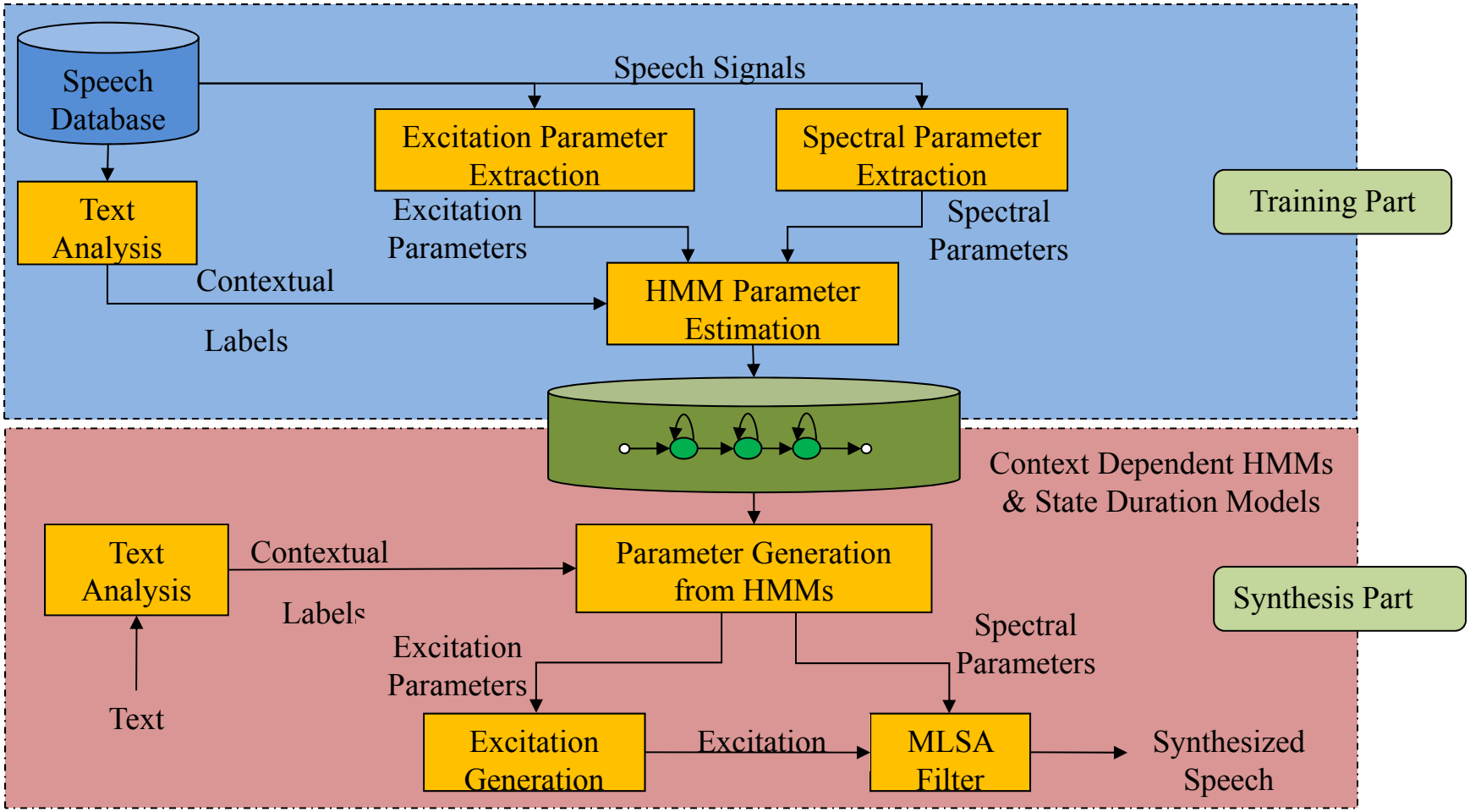
Unit Selection

- The most important issue is to still get a natural and smooth prosody.
- This is hard because the units contain both intonation and pronunciation since entire phrases are used almost directly from the recorded data.
- Since the first Unit selection voice was released, over eight years ago, there has been much improvement for each new voice with every release.

HMM synthesis

- A quite new technology is speech synthesis based on HMM, a mathematical concept called Hidden Markov models.
- It is a statistical method where the text-to-speech system is based on a model that is not known before hand but it is refined by continuous training.
- The technique consumes large CPU resources but very little memory.
- This approach seems to give a better prosody, without glitches, and still produces very natural sounding, human-like speech

HMM-based Speech Synthesis System (HTS)



Automatic Speech Recognition (ASR)

❑ ASR

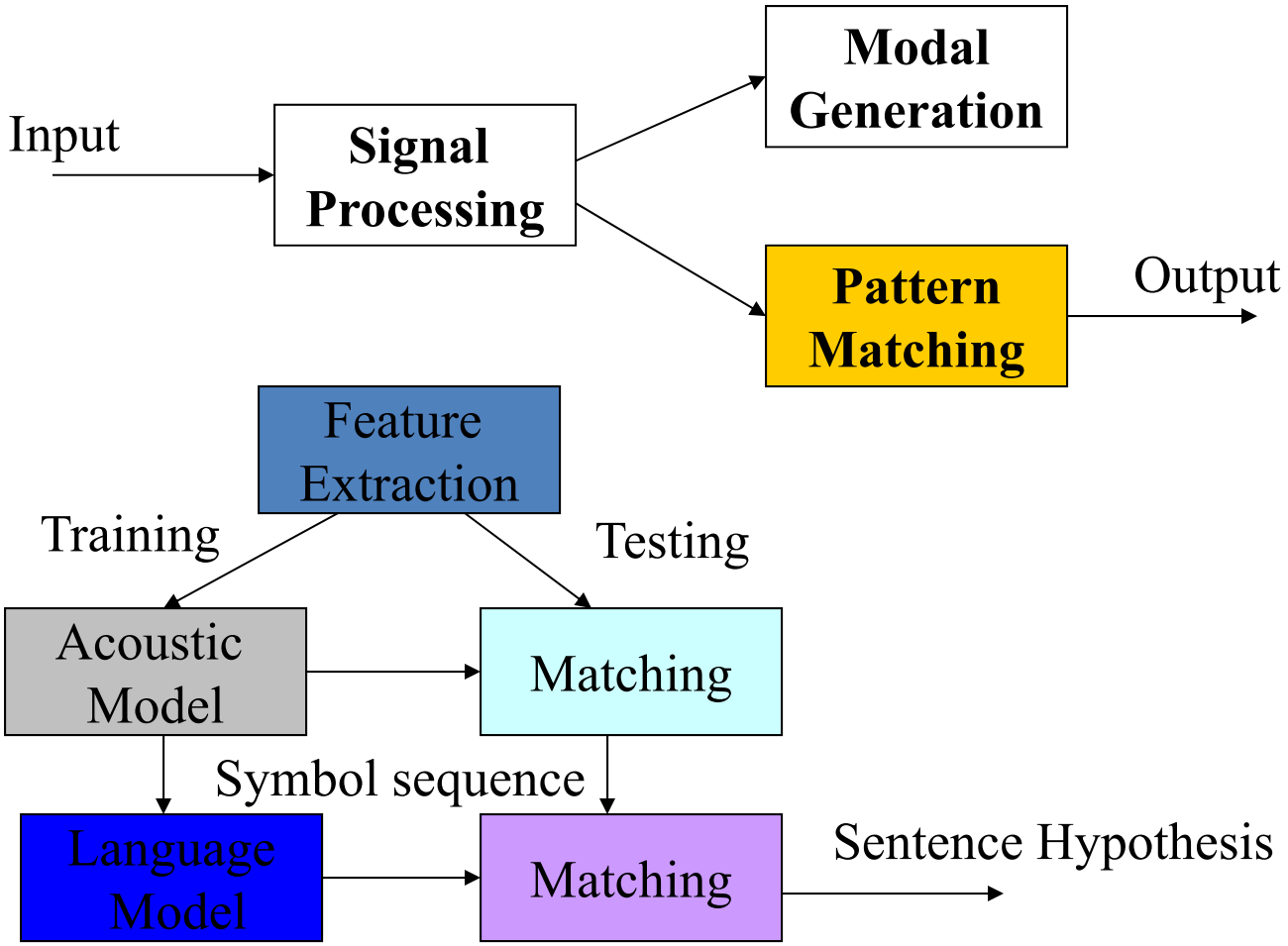
- incoming acoustic waveform ->sequence of linguistic words

Speech is the most natural means of communication for humans and usually learned to a high degree of proficiency at a very early age.

❑ ASR Applications:

- Natural human-machine communication
- Interactive problem solving
- Telecommunications
  - Cost reduction (replace human operators)
  - Revenue generation (new services)
- Hands-free operation
- Dictation & automatic transcription
- Aids for handicapped
- Automatic language translation

Speech Recognition

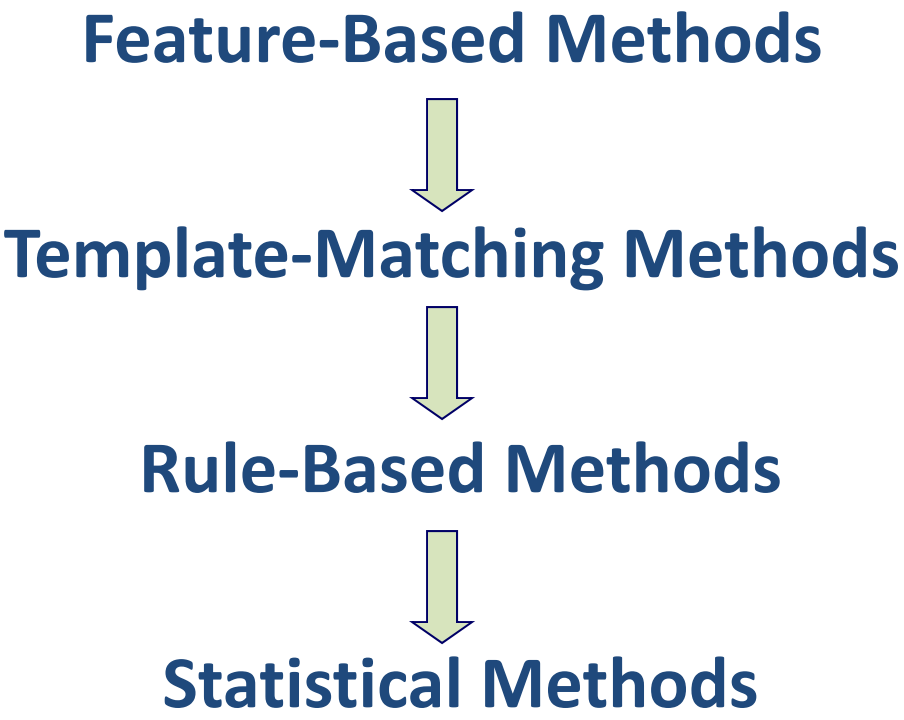


The Speech Recognition Problem

- If we know the signal patterns that represent every spoken word beforehand, we could try to identify the words whose patterns best match the input
  - Word patterns are never reproducible exactly
  - How do we represent these signal patterns?
  - Given this uncertainty, how do we compare the input to known patterns?
- Problem in word pattern
  - Large vocabulary → As vocabulary size increases, complexity increases
  - Absence of word boundary markers in continuous speech
  - Inherent ambiguities: “I scream” or “Ice cream”?

- Tremendous range of variability in speech, even though the message may be constant:
  - Human physiology:
  - Speaking style: clear, spontaneous, etc.
  - Speaking rate: fast or slow speech
  - Emotional state: happy, sad, etc.
  - Emphasis: stressed speech *vs.* unstressed speech
  - Accents, dialects, foreign words
  - Environmental or background noise

Trends in the Developments of ASR



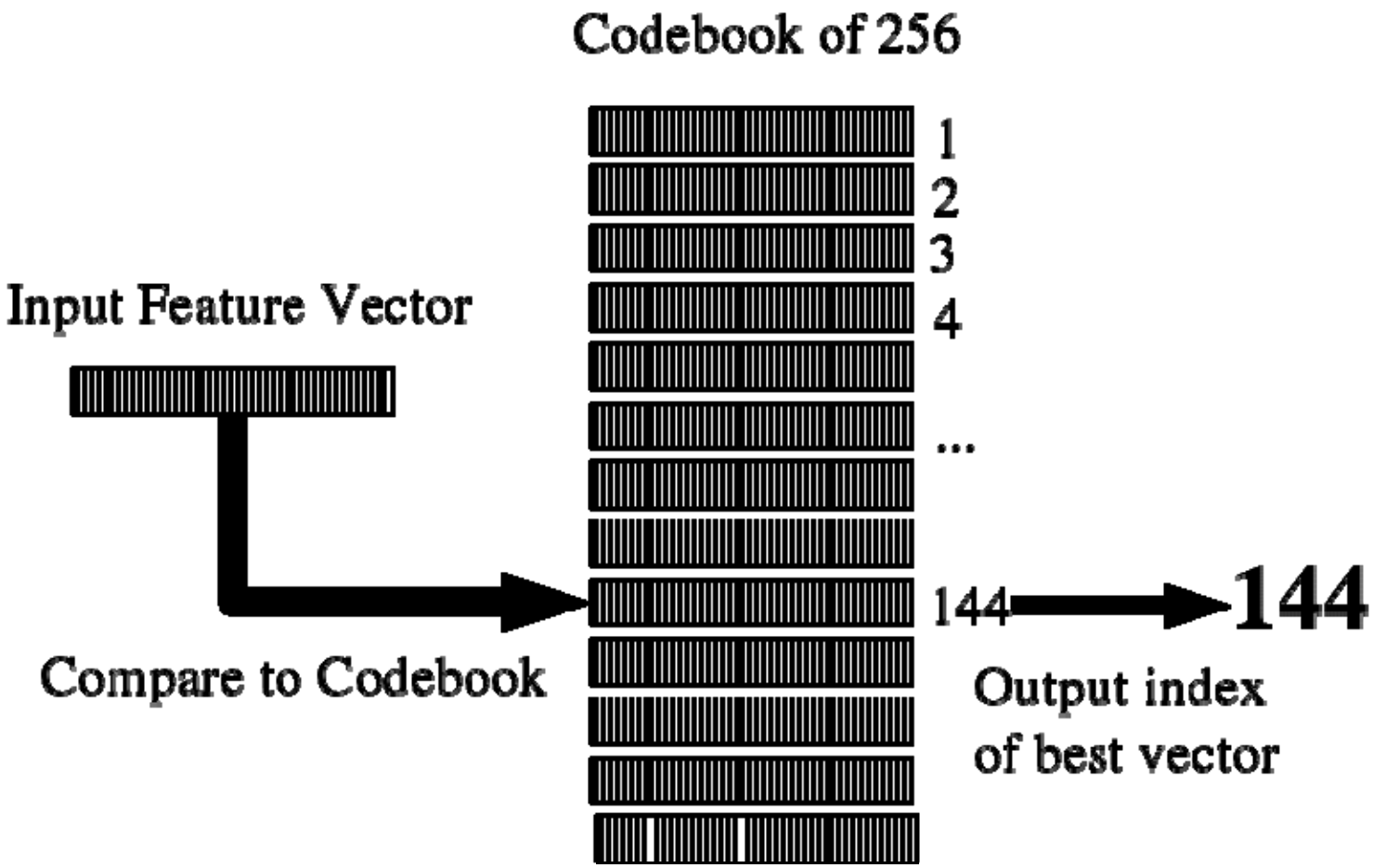
ASR: Pattern Matching Methods

- **Vector Quantization (VQ)**
  - Code book Models are used
  - A VQ codebook is designed by standard clustering procedures for each enrolled speaker using his training data, usually based upon reading a specific text.
  - The pattern match score is the distance between an input vector and the minimum distance code word in the VQ codebook C
- **Dynamic Time Warping (DTW)**
  - Template Models are used
  - A text-dependent template model is a sequence of templates (X1 ..XN) that must be matched to an input sequence (Y1 ..YM).
- **Hidden Markov Models (HMM)**
  - Statistical Models are used
  - Using a stochastic model, the pattern-matching problem can be formulated as measuring the likelihood of an observation (a feature vector of a collection of vectors from the unknown speaker) given the speaker model

Vector Quantization

- Create a training set of feature vectors
- Cluster them into a small number of classes
- Represent each class by a discrete symbol
- We'll define a
  - Codebook, which lists for each symbol
  - A prototype vector, or codeword
- If we had 256 classes ('8-bit VQ'),
  - A codebook with 256 prototype vectors
  - Given an incoming feature vector, we compare it to each of the 256 prototype vectors
  - We pick whichever one is closest (by some 'distance metric')
  - And replace the input vector by the index of this prototype vector

VQ

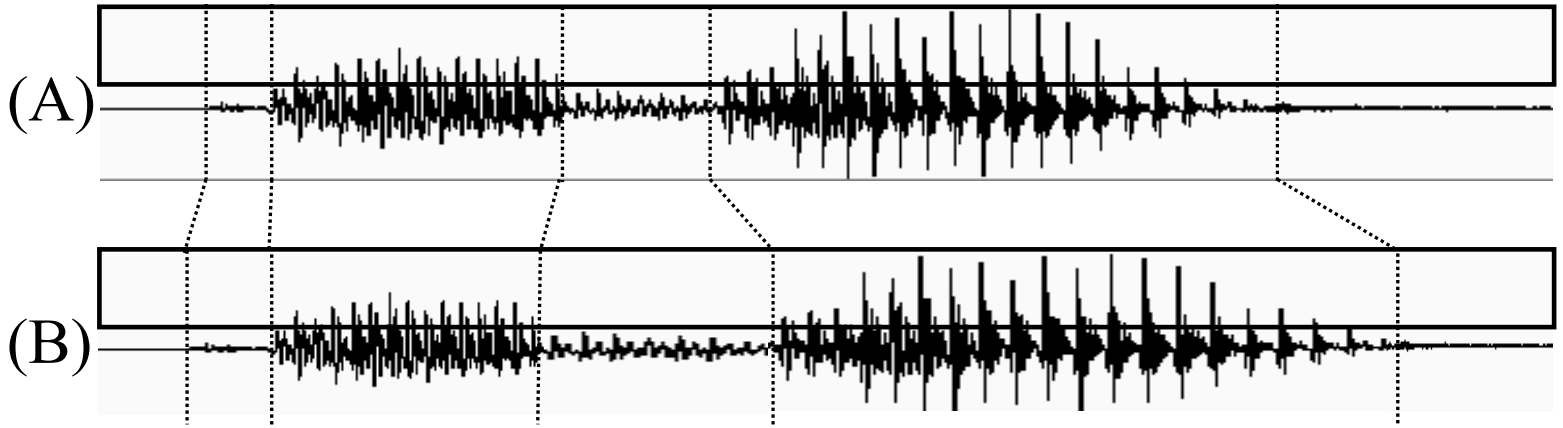

Codebook of 256
1
2
3
4
...
144
144
Input Feature Vector
Compare to Codebook
Output index
of best vector

VQ requirements

- A distance metric or distortion metric
  - Specifies how similar two vectors are
  - Used:
    - to build clusters
    - To find prototype vector for cluster
    - And to compare incoming vector to prototypes
- A clustering algorithm
  - K-means, etc.

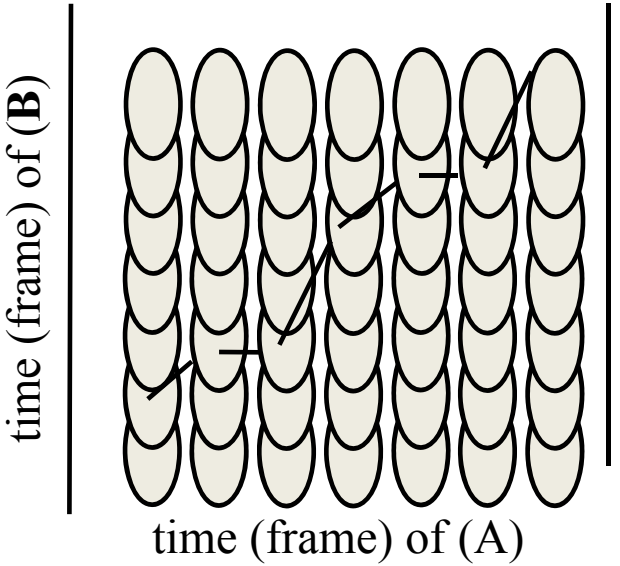
Dynamic Time Warping (DTW)

- **Goal:** Given two utterances, find "best" alignment between pairs of frames from each utterance.



The path through this matrix shows the best pairing of frames from utterance A with utterance B:

This path can be considered the best "warping" between A and B.



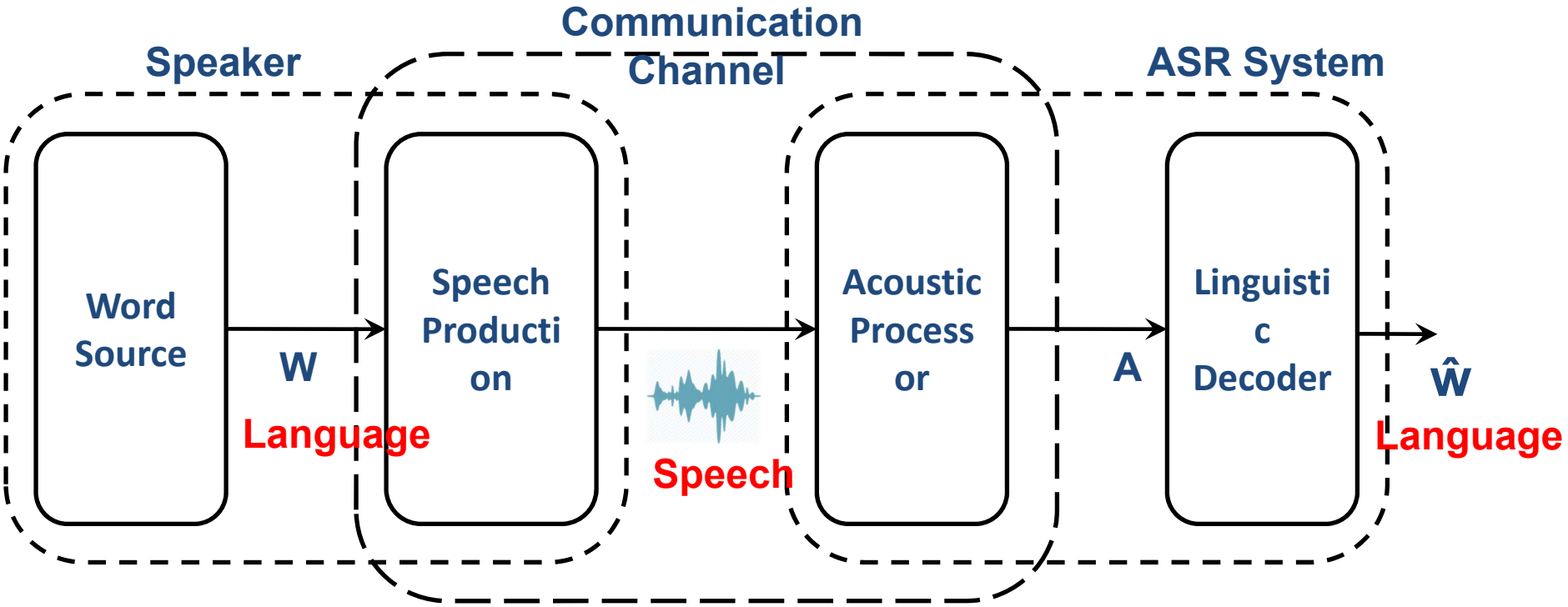
Dynamic Time Warping (DTW)

- Dynamic Time Warping
  - Requires measure of "distance" between 2 frames of speech, one frame from utterance A and one from utterance B.
  - Requires heuristics about allowable transitions from one frame in A to another frame in A (and likewise for B).
  - Uses inductive algorithm to find best warping.
  - Can get total "distortion score" for best warped path.
- Distance:
  - Measure of dissimilarity of two frames of speech
- Heuristics:
  - Constrain begin and end times to be (1,1) and (T,T)
  - Allow only monotonically increasing time
  - Don't allow too many frames to be skipped
  - Can express in terms of "paths" with "slope weights"

Dynamic Time Warping (DTW)

- Does not require that both patterns have the same length
- We may refer to one speech pattern as the “input” and the other speech pattern as the “template”, and compare input with template.
- For speech, we divide speech signal into equally-spaced frames (e.g. 10 msec) and compute one set of features per frame. The local distance measure is the distance between features at a pair of frames (one from A, one from B).
- Local distance between frames called *d*. Global distortion from beginning of utterance until current pair of frames called *D*.
- DTW can also be applied to related speech problems, such as matching up two similar sequences of phonemes.

State-of-the-art: The Statistical Model



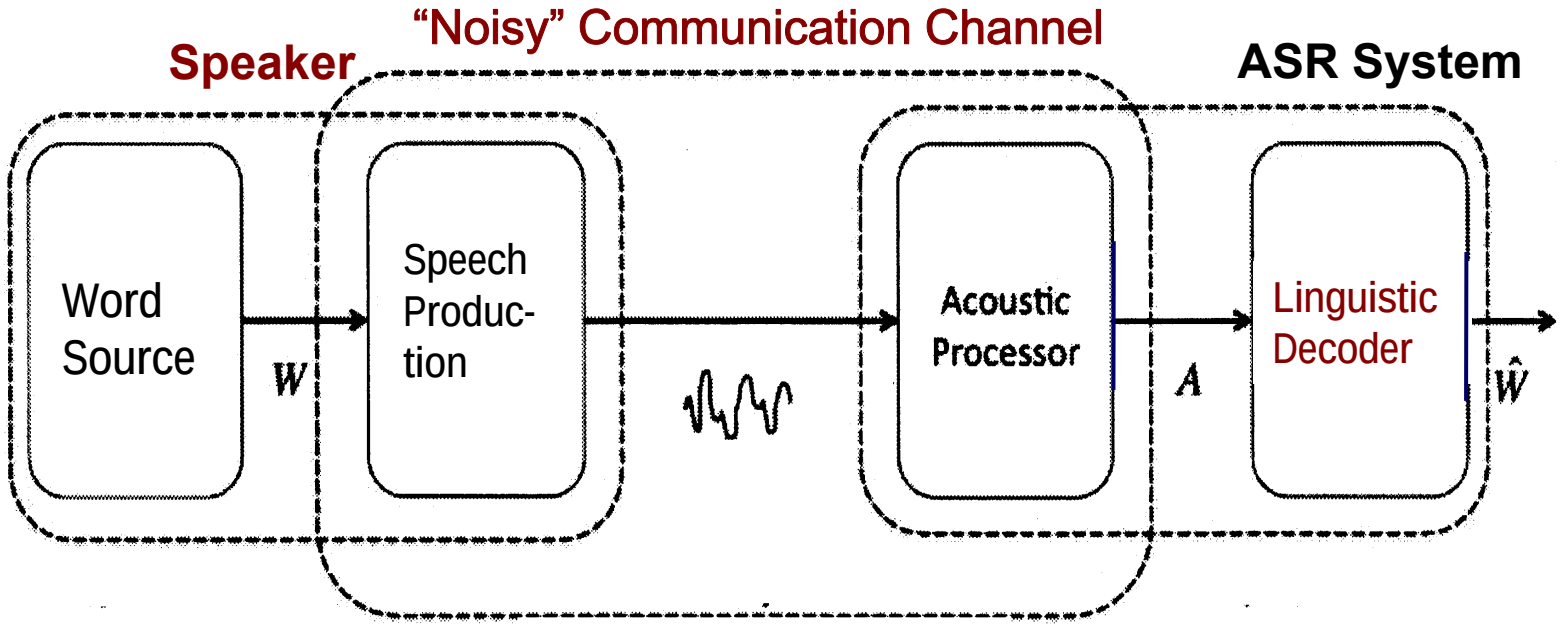
W : a string of words
A : acoustic signal observed by the recognizer
S : set of sentences that can be generated

$$\hat{W} = \arg\max_{W \in S} P(W|A) = \arg\max_{W \in S} P(A|W)\,P(W)$$

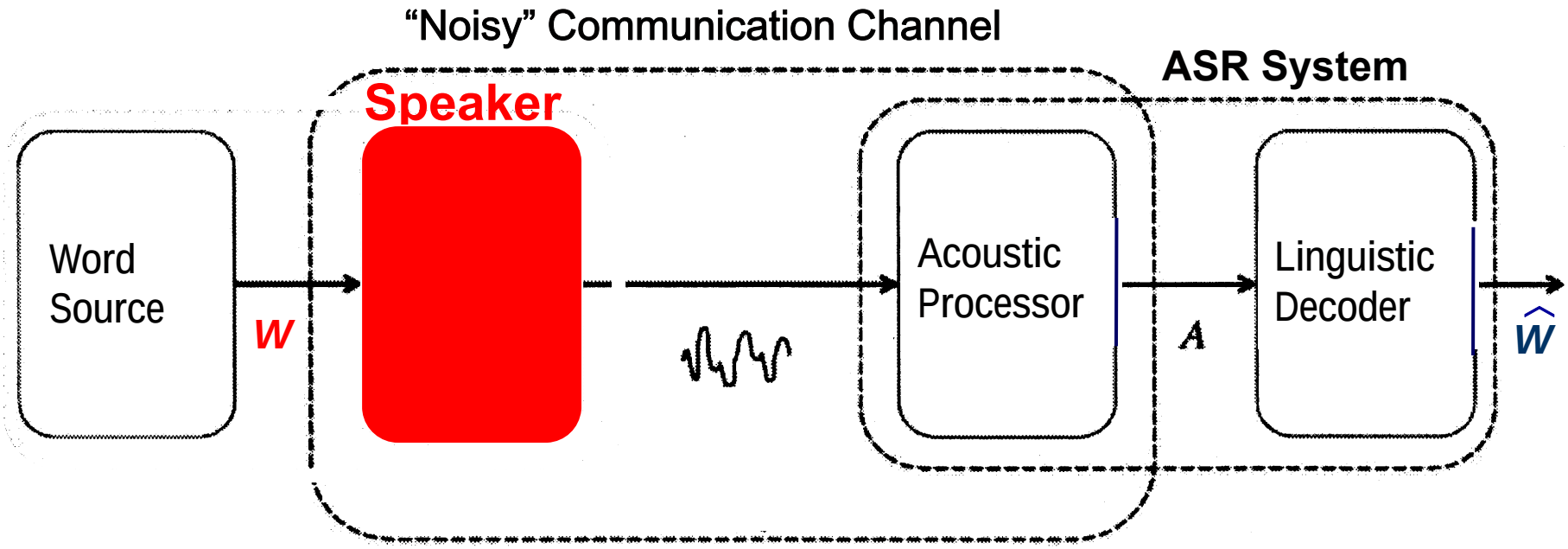
P(A|W): the acoustic model

P(W): the language model

Ref: Jelinek, 1991

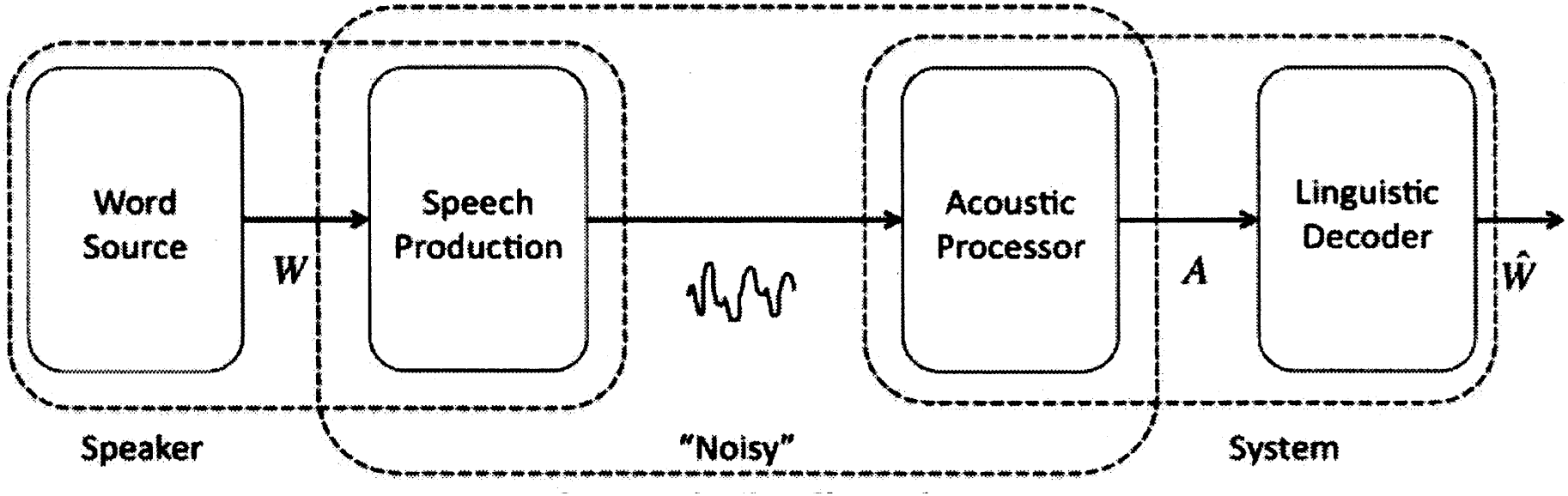


1. Is the speaker model true?
2. Is the noisy communication channel model valid?
3. Is the language model sufficiently accurate?



Since $\hat{W}$ is the estimated output (string of words in the ***written language***), the input ***W*** *must* also be a message in the ***written language***. Hence this scheme shows a situation where the speaker is given a **text** (by someone else) and reads it aloud.

What does a speaker do?



The speaker's functions are:

1. **Understanding** the input message ***W in the written language.***
   (Reading *W* aloud is possible only after understanding it.)
2. **Generating the message *S* *in the spoken language.***

Is the language model sufficiently accurate?

Since a bigram (simple Markov) model is regarded insufficient for representing syntactic relationships between words, it has become a common practice to use trigram (second-order Markov) model as the language model. But trigram is still not good enough.

Simple Markov Model = Bigram Model

$$\hat{P}(W) = P(w_1, w_2, w_3, ..., w_k)$$
$$\approx \hat{P}(w_1)\hat{P}(w_2 \mid w_1)\hat{P}(w_3 \mid w_2).....\hat{P}(w_k \mid w_{k-1})$$

For a thousand word vocabulary, measurement of bigram probability needs to examine occurrences of each of one million bigrams.

$$\hat{P}(W) = P(w_1, w_2, w_3, ..., w_k)$$
$$\approx \hat{P}(w_1 w_2)\hat{P}(w_3 \mid w_1 w_2)\hat{P}(w_4 \mid w_2 w_3).....\hat{P}(w_k \mid w_{k-2} w_{k-1})$$

For a thousand word vocabulary, measurement of trigram probability needs to examine occurrences of each of one billion trigrams. Since data is not enough, smoothing is introduced, but the validity is not guaranteed. Still, trigram is a very poor model for syntax.

Limitations of the Statistical Approach

The statistical approach is limited since

(1) It does not make distinction between the spoken language and the written language

(2) The noisy communication channel model does not approximate the situation involving ambiguity

(3) It is impossible to use higher-order statistics beyond word trigrams.

Spoken Language vs. Written Language

Conventional Concepts: Language vs. Speech

- "Language" is the underlying code system common to both speech and text.
- "Speech" is the acoustic signal carrying the information of the language.

New Concepts (Fujisaki, 1986) :

Spoken Language vs. Written Language

- "Speech" and "Text" are physical signals but also serve as different code systems. **The information they carry are not identical.**
- "Spoken Language" refers to both the signal and the code system. The same applies to "Written Language."

Examples of Differences Between SL and WL as Code Systems

1. Difference in ambiguity:
   Many expressions exist that are ambiguous in WL but are unambiguous in SL, and *vice versa*; e.g., homonymity and homographism at the lexical level.

   Homographism (ambiguous in WL but not in SL)
   English: close, content, interest, increase, present, record, recreation, wind, etc.

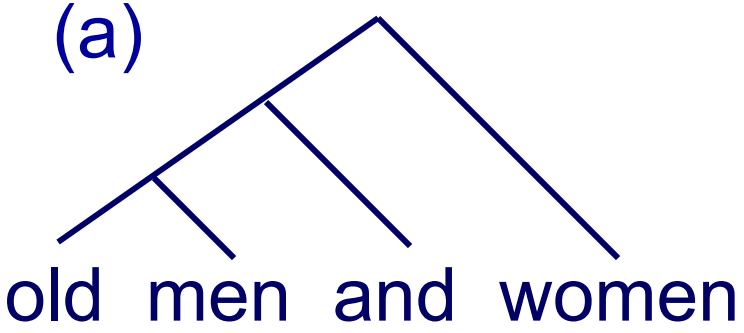
   Homonymity (ambiguous in SL but not in WL)
   English: flower/flour, knight/night, made/maid, pray/prey, right/write, steal/steel, etc.

Examples of Differences Between SL and WL as Code Systems

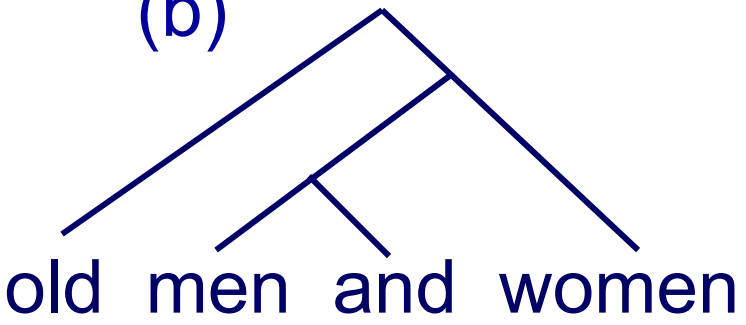
2. Prosody in SLs allows disambiguation of some of the ambiguities (e.g., lexical and syntactic) in WLs.

Example: syntactic ambiguity in WL

(a)



(b)



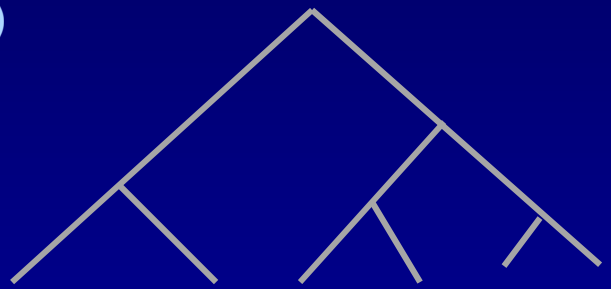
(a) All the young men went to war, leaving only old men and women at home.

(b) Old men and women are called senior citizens.

- **Prosody in Spoken Languages allows disambiguation of some of the lexical and syntactic ambiguities in Written Languages.**

Example: syntactic ambiguity in WL

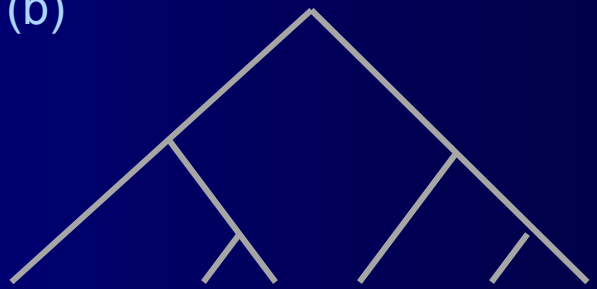
(a)



পড়াশোনা কর না পড়লে শিখবে না

(unless you study, you can't learn)

(b)



পড়াশোনা কর না পড়লে শিখবে না

(if you study, you can't learn)

Examples of Differences Between SL and WL as Code Systems

3. Difference in word boundary marking

   In many (but not all) WLs, word boundaries are explicitly shown, but not in SLs, causing ambiguity in SL.

   English: I scream / ice cream ➡ [aiskri:m]

   an ice cream / a nice cream ➡ [ənaiskri:m]

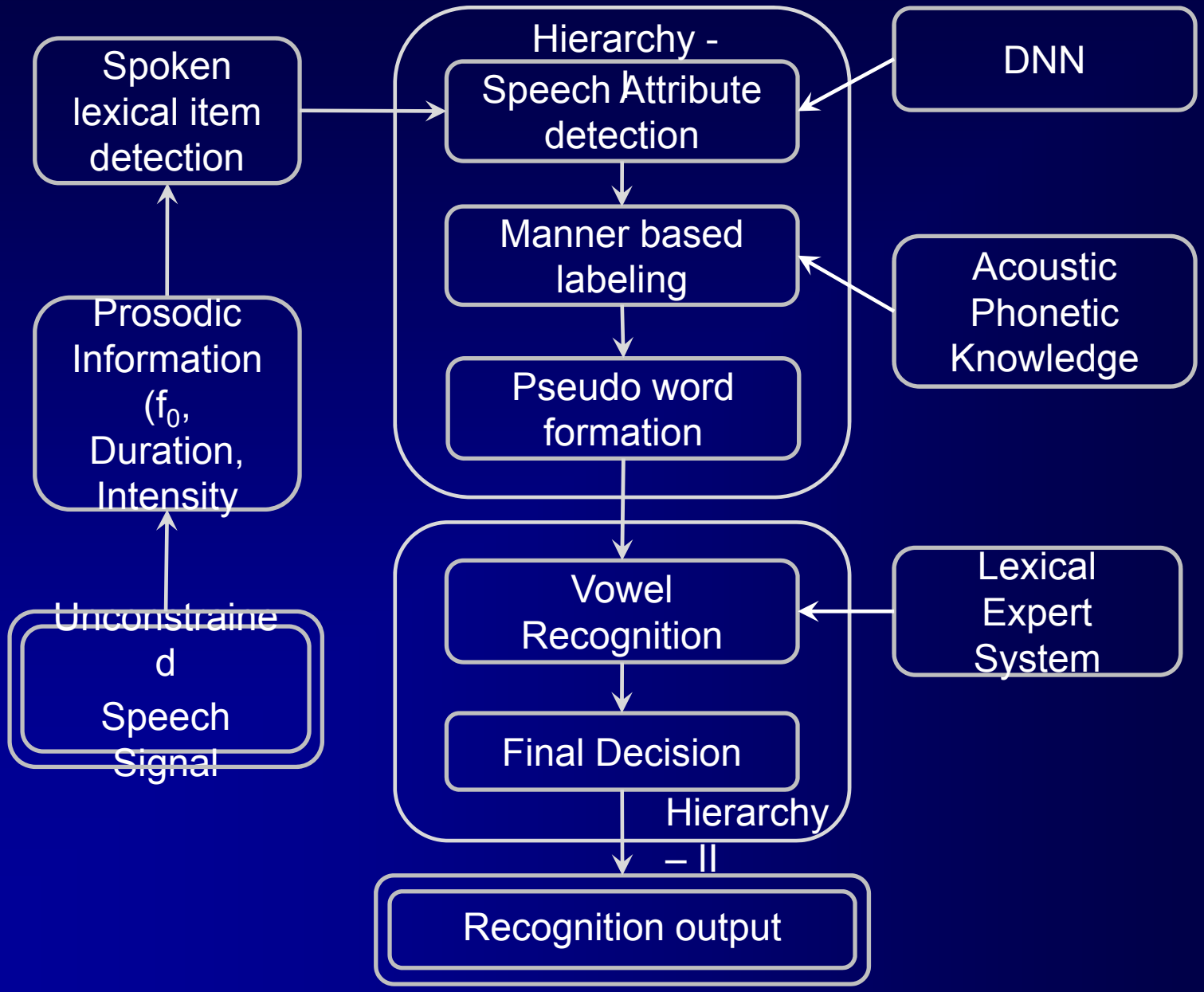
   night rate / niterate ➡ [naitreit]

SL and WL are Different Code Systems

- Certain kind/amount of **linguistic** information is **NOT** shared by the Spoken Language (SL) and the Written Language (WL).
- Hence they are two distinct systems for coding thelinguistic information.
- Conversion of a message expressed in SL into a message in WL (*i.e.*, ASR) is only possible by correctly inferring the information missing in the SL message, and by expressing it in the WL message.
- The same is true for conversion of a WL message into an SL message.

Spoken word based or prosodic word based continuous speech recognition system.

Proposed Model



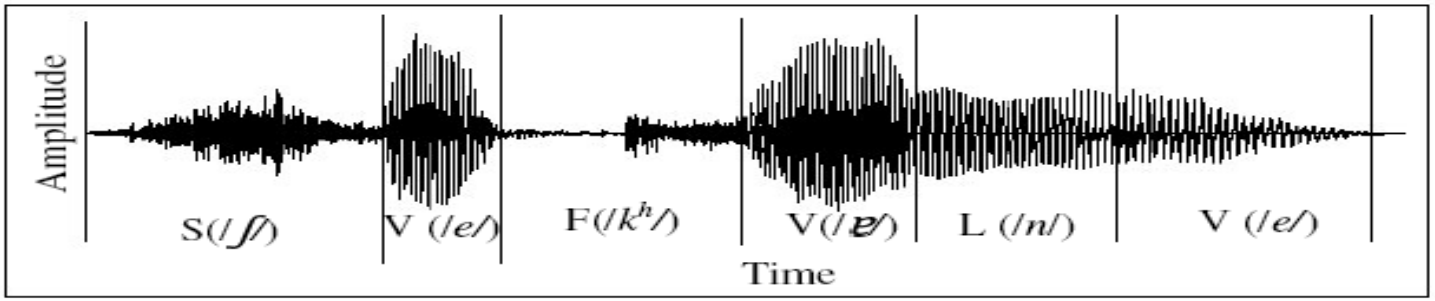
Final selected manner for pseudo words generation

Sl no.	Manner	Bangla phonemes for manner base labeling
1	S	/ʃ/, /s/
2	P	/k/, /t/, /ʈ/, /p/
3	F	/k^h/, /t^h/, /$ʈ^h$/, /p^h/, /ʧ/, /ʧ^h/
4	A	/g/, /d/, /ɖ/, /b/, /g^h/, /d^h/, /$ɖ^h$/, /b^h/, /ʤ/, /ʤ^h/
5	L	/l/, /m/, /n/
6	V	/ɔ/, /ɐ/, /i/, /u/, /e/, /o/, /æ/

Pseudo word

If the pronunciation of a word is represented by the selected manner symbols then the new representation of the word is called pseudo word

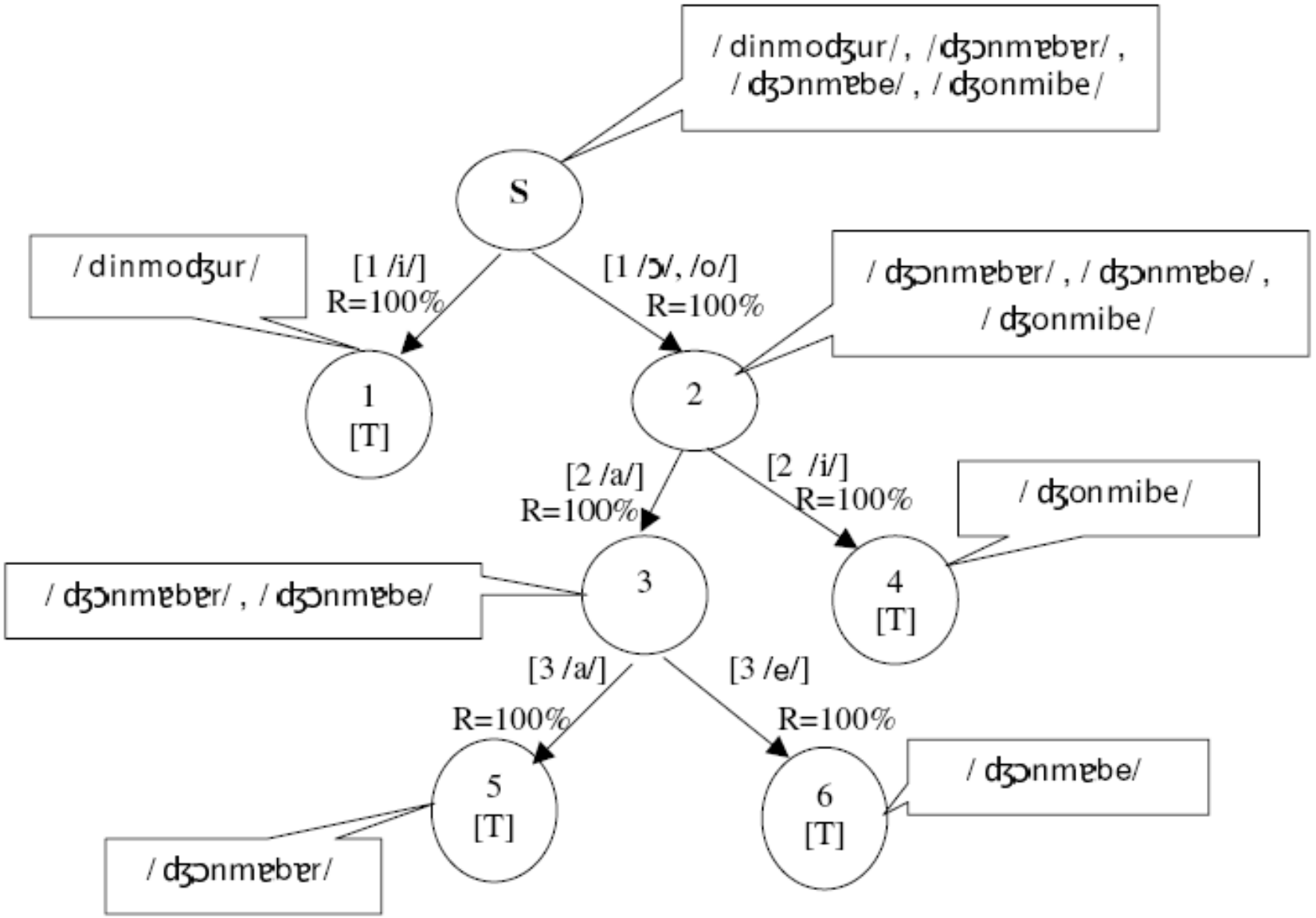
As for example if the word is / ʃekʰɐne/ then the corresponding pseudo word after manner based labeling is SVFVLV

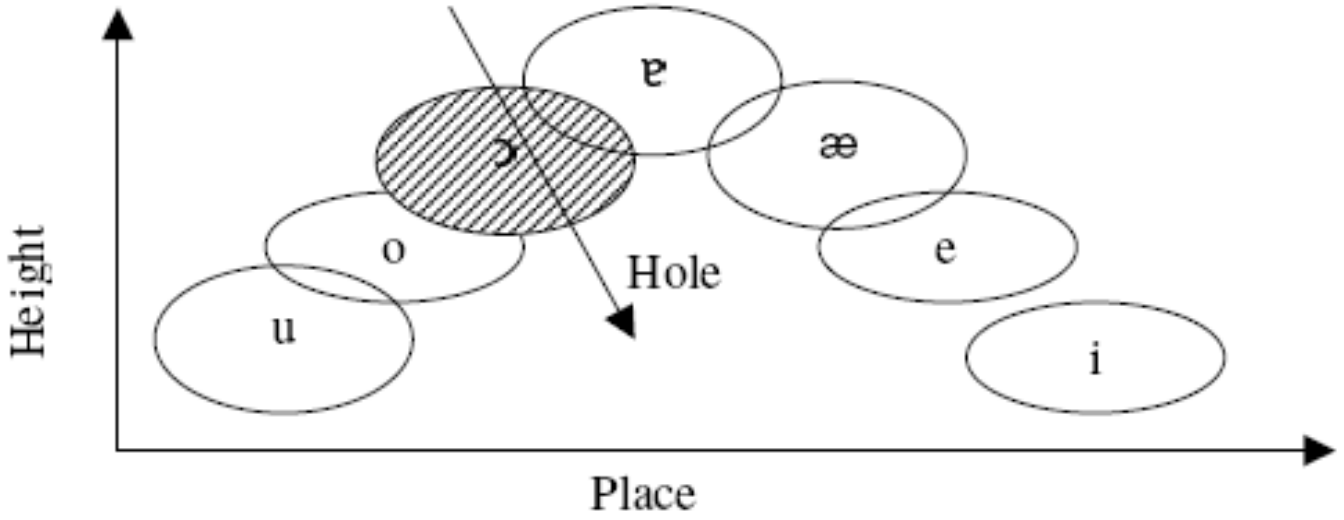


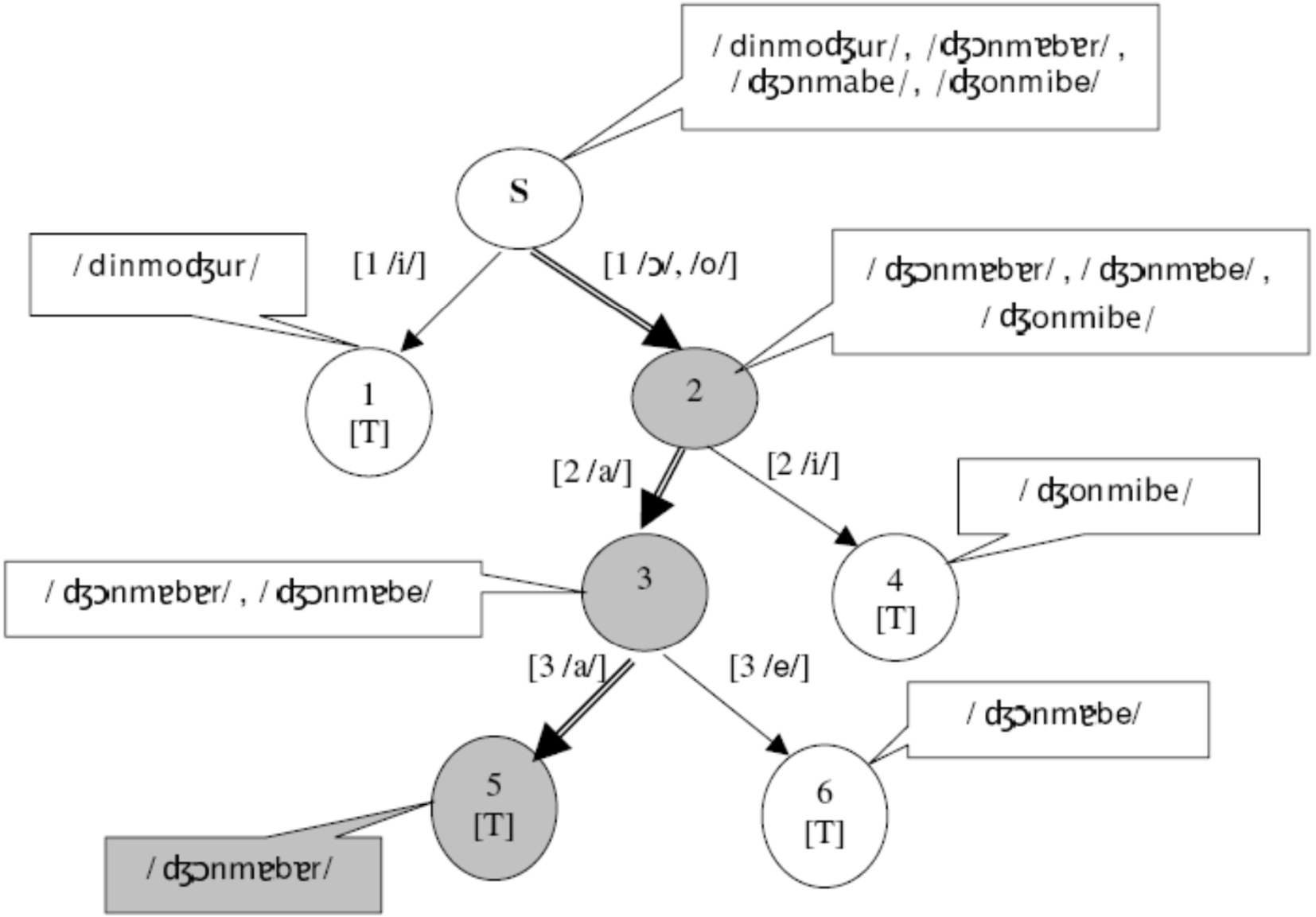
Cohorts

A group of words of same pseudo word representation forms a cohort. For Example cohort AVLLVAV consists of four Bangla words, /dinmodʒur/, /dʒɔnmɐbɐr/, /dʒɔnmɐbe/, /dʒonmibe/.

দিনমজুর, জন্মাবার, জন্মাবে, জন্মিবে


/ dinmodʒur/ , /dʒɔnmɐbɐr/ ,
/ dʒɔnmɐbe/ , / dʒonmibe/
S
/ dinmodʒur/
[1 /i/]
R=100%
[1 /ɔ/, /o/]
R=100%
/ dʒɔnmɐbɐr/ , / dʒɔnmɐbe/ ,
/ dʒonmibe/
1
[T]
2
[2 /a/]
R=100%
[2 /i/]
R=100%
/ dʒonmibe/
/ dʒɔnmɐbɐr/ , / dʒɔnmɐbe/
3
4
[T]
[3 /a/]
R=100%
[3 /e/]
R=100%
5
[T]
6
[T]
/ dʒɔnmɐbe/
/ dʒɔnmɐbɐr/


Height
ɐ
ɔ
æ
o
Hole
e
u
i
Place


/ dinmodʒur/, /dʒɔnmɐbɐr/ ,
/ dʒɔnmabe/, /dʒonmibe/
S
/ dinmodʒur/
[1 /i/]
[1 /ɔ/, /o/]
/ dʒɔnmɐbɐr/ , / dʒɔnmɐbe/ ,
/ dʒonmibe/
1
[T]
2
[2 /a/]
[2 /i/]
/ dʒonmibe/
/ dʒɔnmɐbɐr/ , / dʒɔnmɐbe/
3
4
[T]
[3 /a/]
[3 /e/]
/ dʒɔnmɐbe/
5
[T]
6
[T]
/ dʒɔnmɐbɐr/

Speech based Technology Development for e-Learning

Area of research

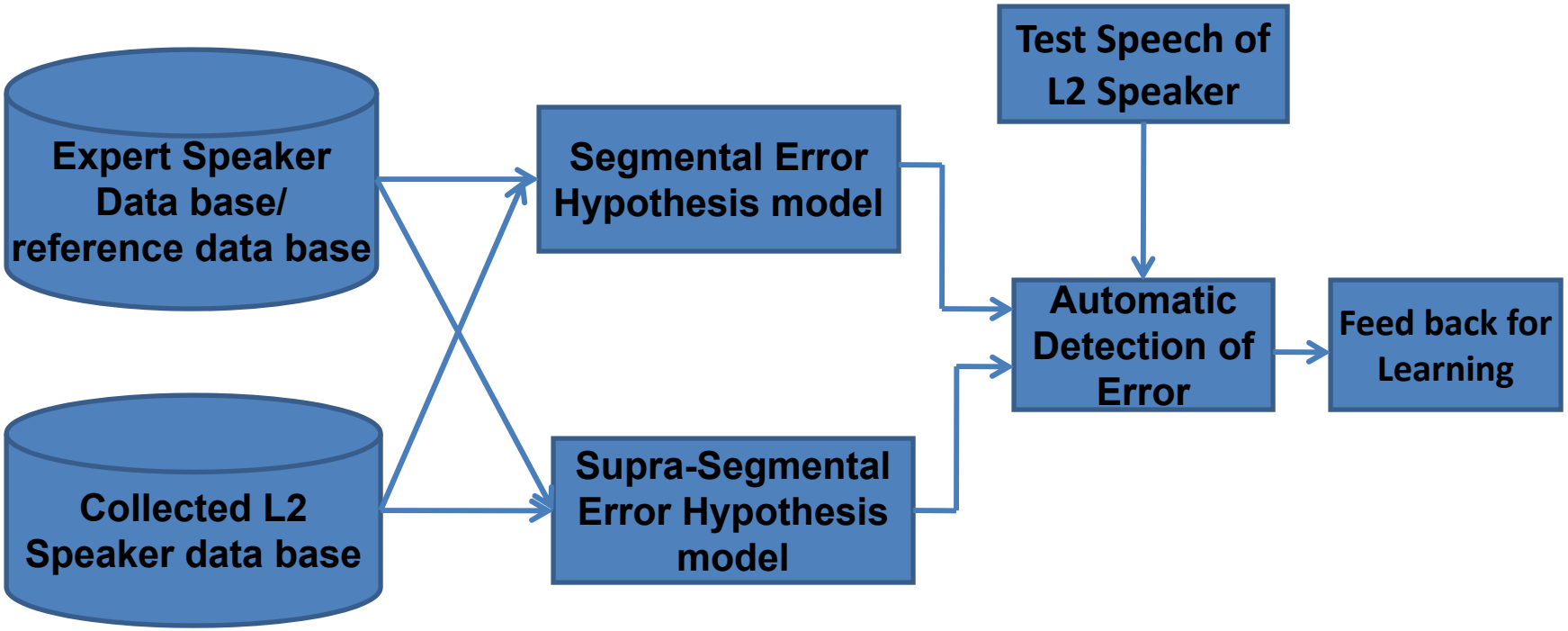
❖ Computer Assisted Spoken Language Learning:

Objective: To Develop a Computer Assisted Spoken Language Learning tools with correct language pedagogy for faster accusation of second language speaking

❖ Accent Conversion:

Objective: Transform the regional/foreign accent of a source speaker to have similar characteristics to that of another accent

Schematic diagram of the Computer Assisted Spoken Language Learning system



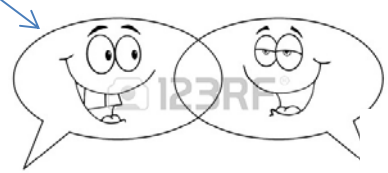
Very good lecture

Source L1 Japanese speaking L2 English

L1 Japanese student

L1 American student

Accent conversion



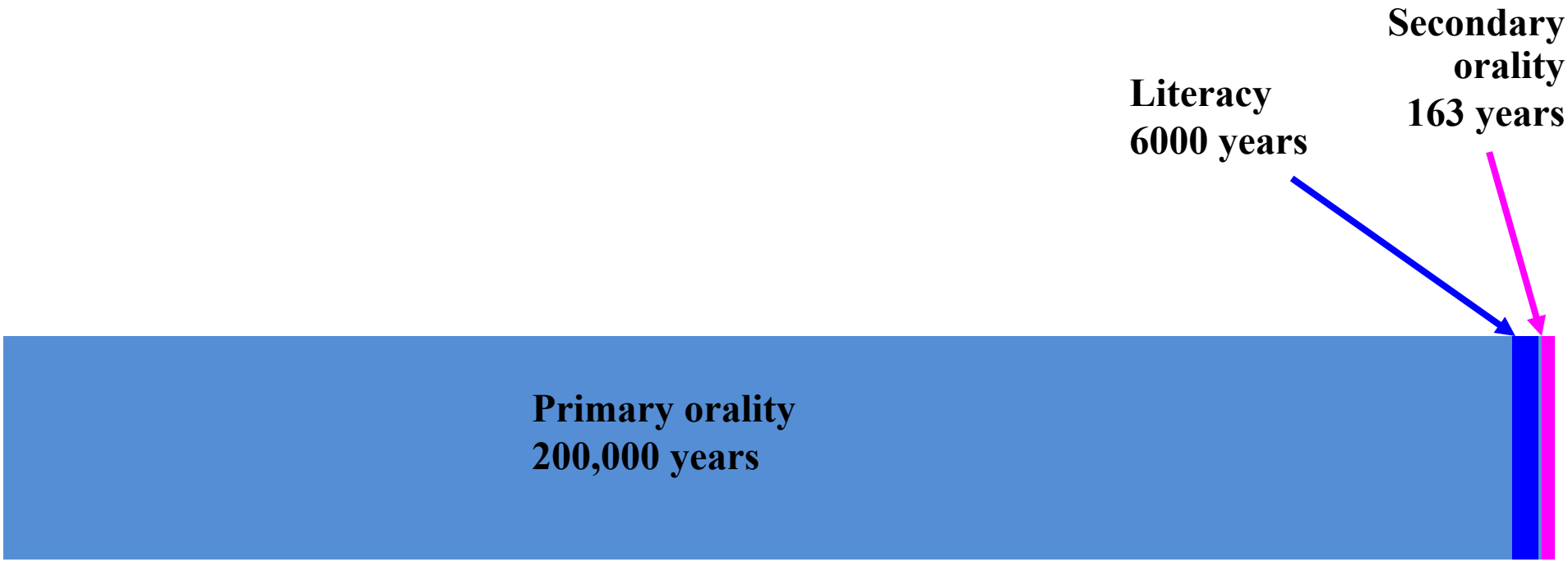
Speech spectacle

???

This is awesome!

Walter Ong, 1982

Orality and Literacy: The Technologizing of the Word



Orality and writing production

Kellogg : Sentence Production Demands: Verbal Working Memory

"Orthographic as well as phonological representations must be activated for written spelling."

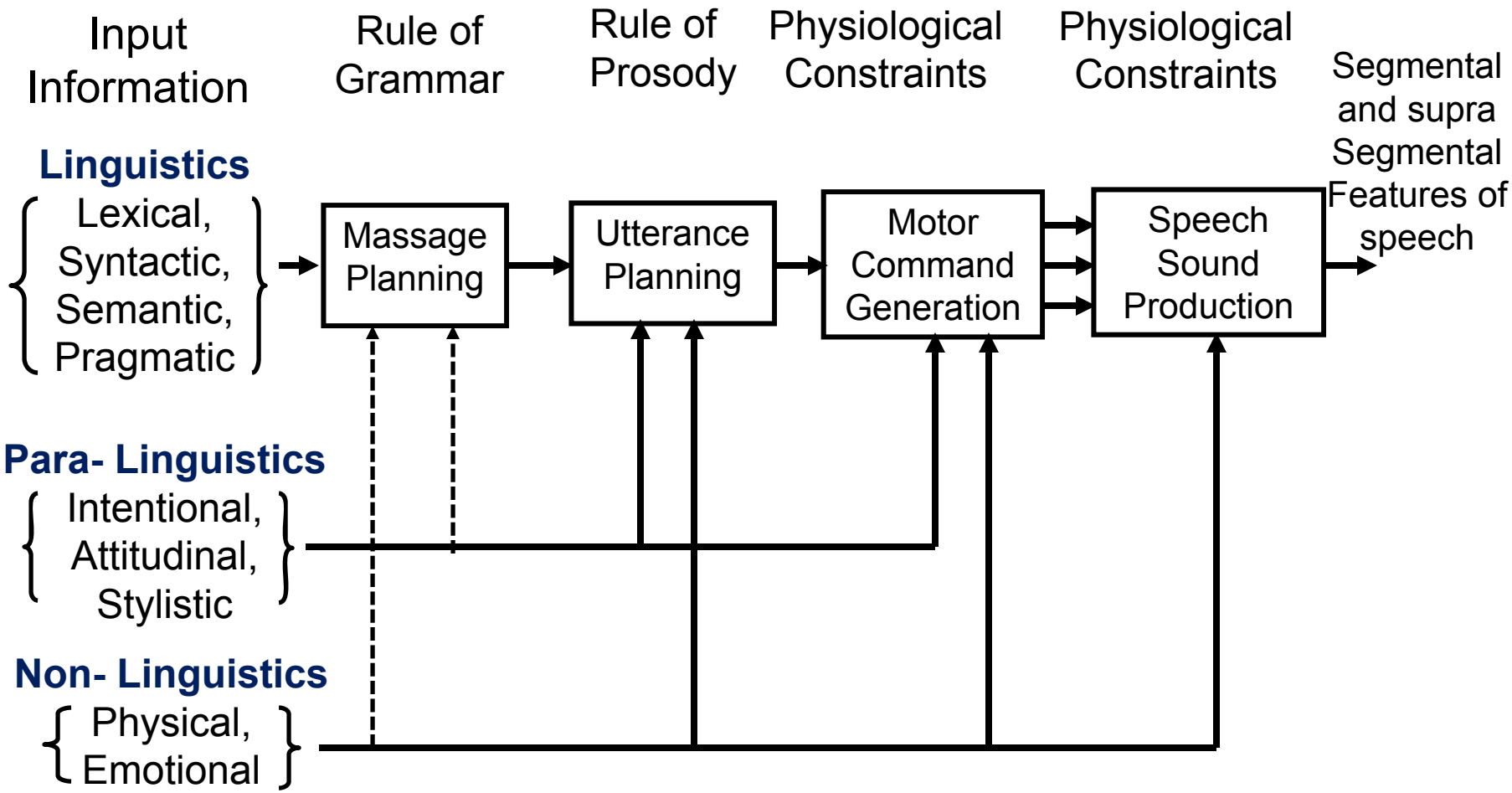
- Bonin, Fayol, & Gombert (1997)

"Verbal WM is necessary to maintain representations during grammatical, phonological, and orthographic encoding."

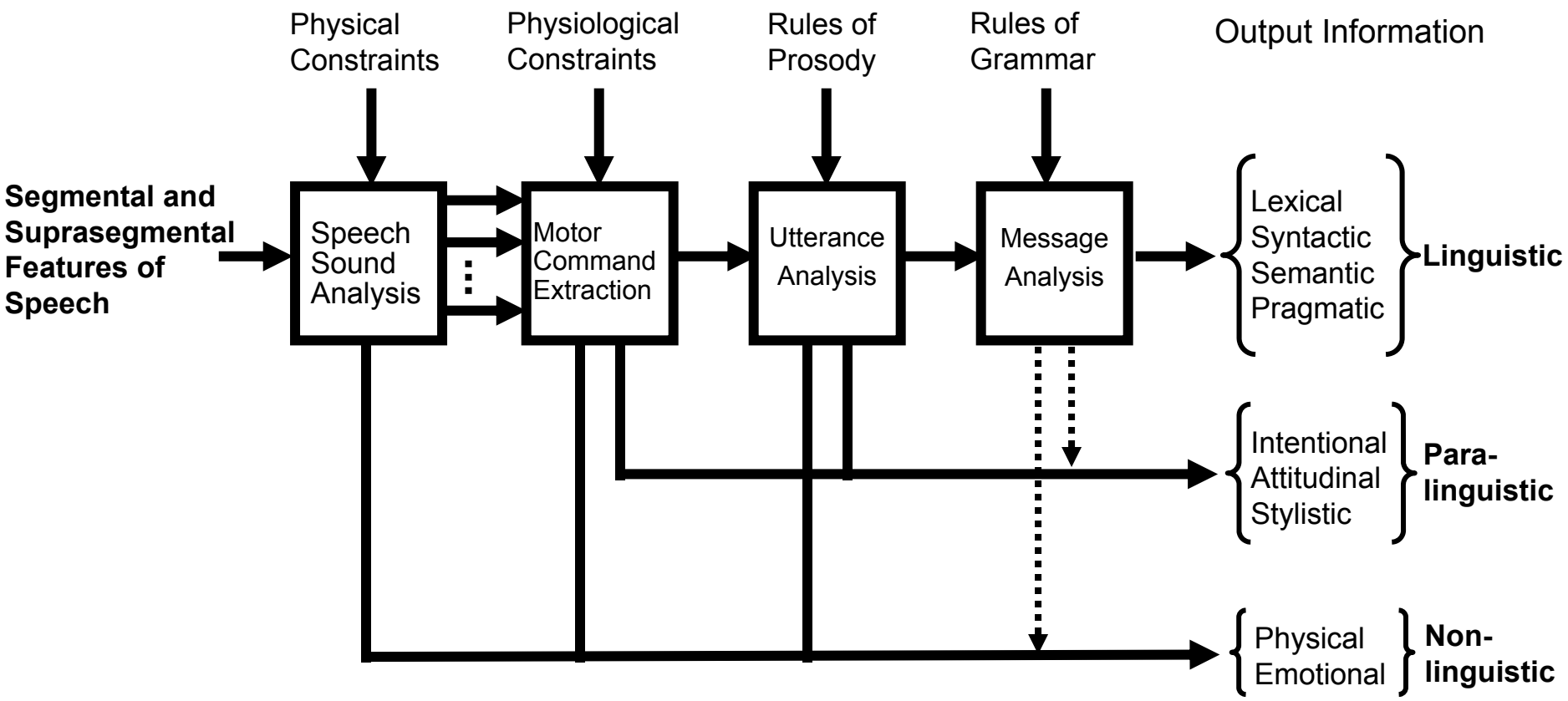
- Levy & Marek (1999)
- Chenoweth & Hayes (2001)
- Kellogg, Olive, & Piolat (2006)

Kellogg, R. (2006) *Training writing skills: A cognitive developmental perspective*. EARLI SigWriting 2006 Antwerp.
http://webhost.ua.ac.be/sigwriting2006/Kellogg_SigWriting2006.pdf

Information manifestation in the segmental and suprasegmental features of speech

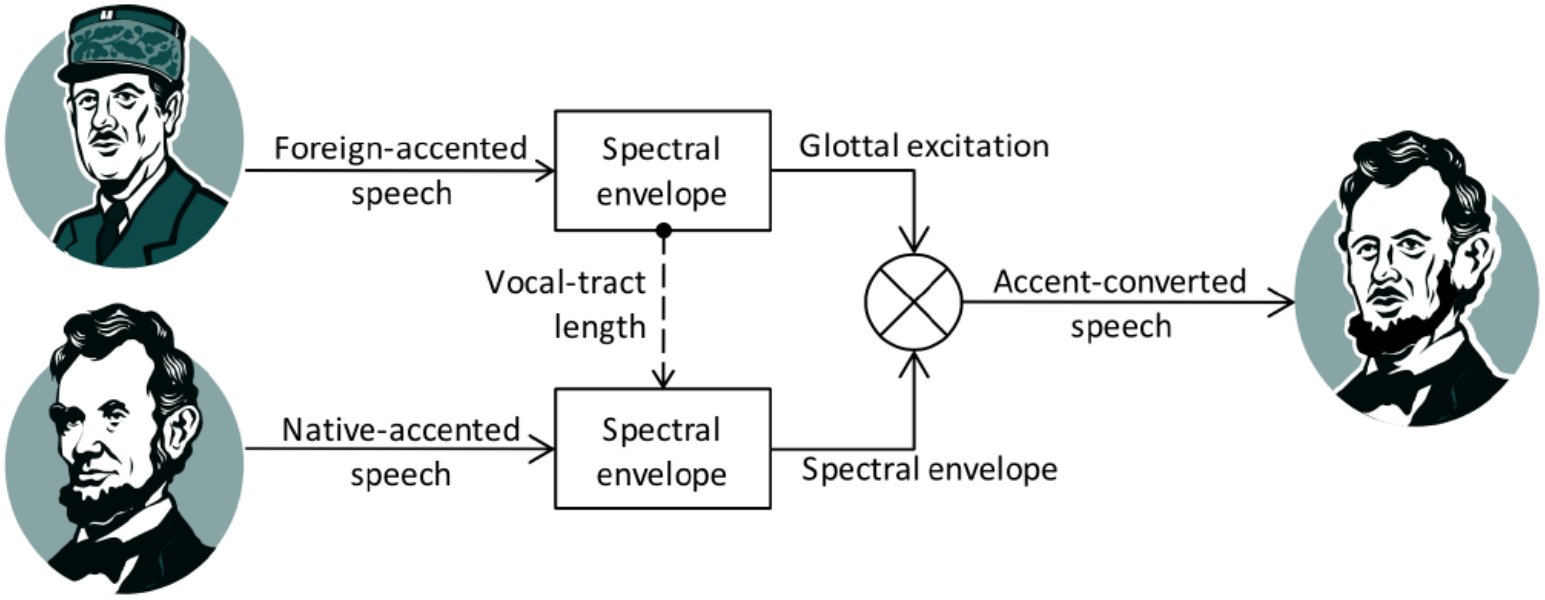


Processes of Information Extraction from Speech



Accent conversion

Idea: Combine the spectral envelope of a native speaker with the excitation and vocal tract length of a non-native speaker



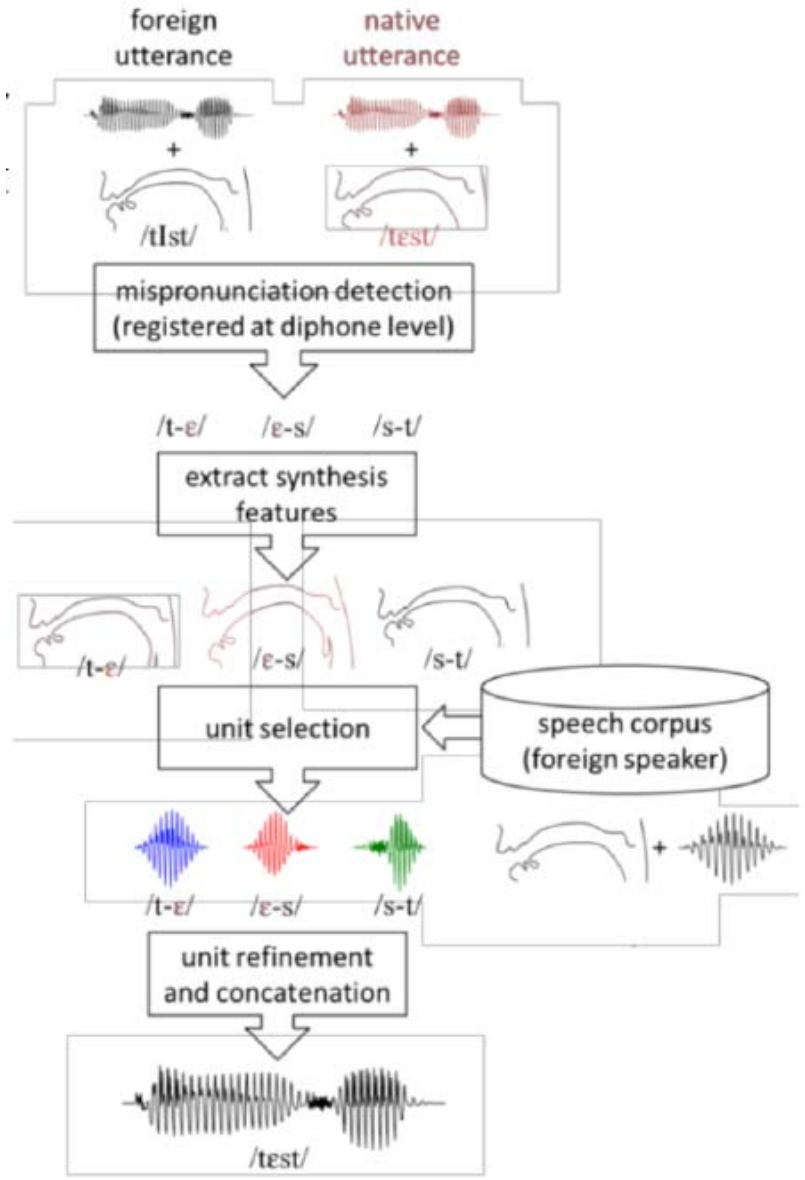
Research Trend

Foreign Accent Conversion Through Concatenative Synthesis in the Articulatory Domain

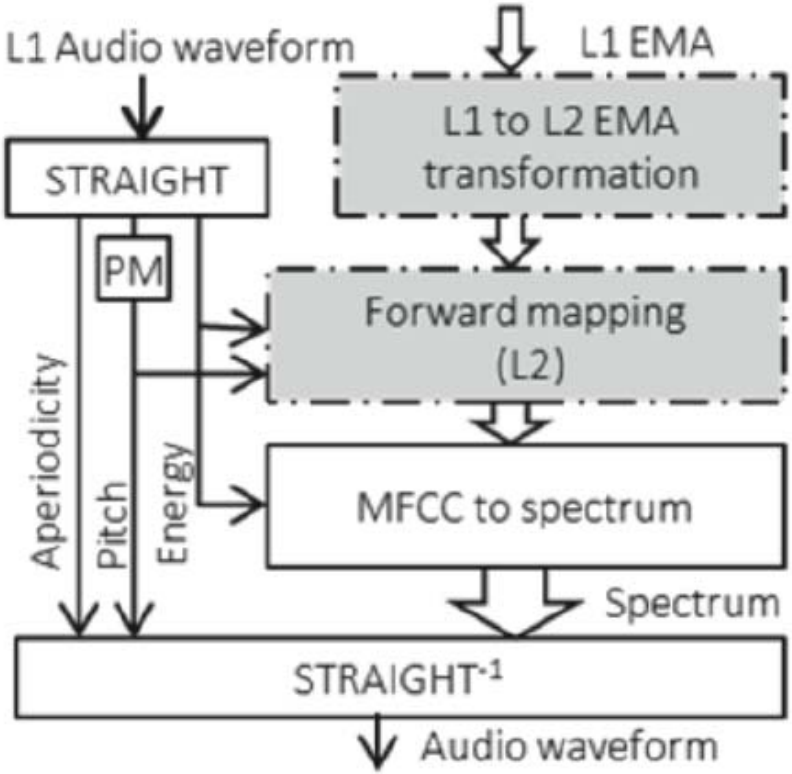
/træb^{h}lar/

t	r	æ	b^{h}	l	a	r	
t	r	æ	b^{h}	e	l	a	r

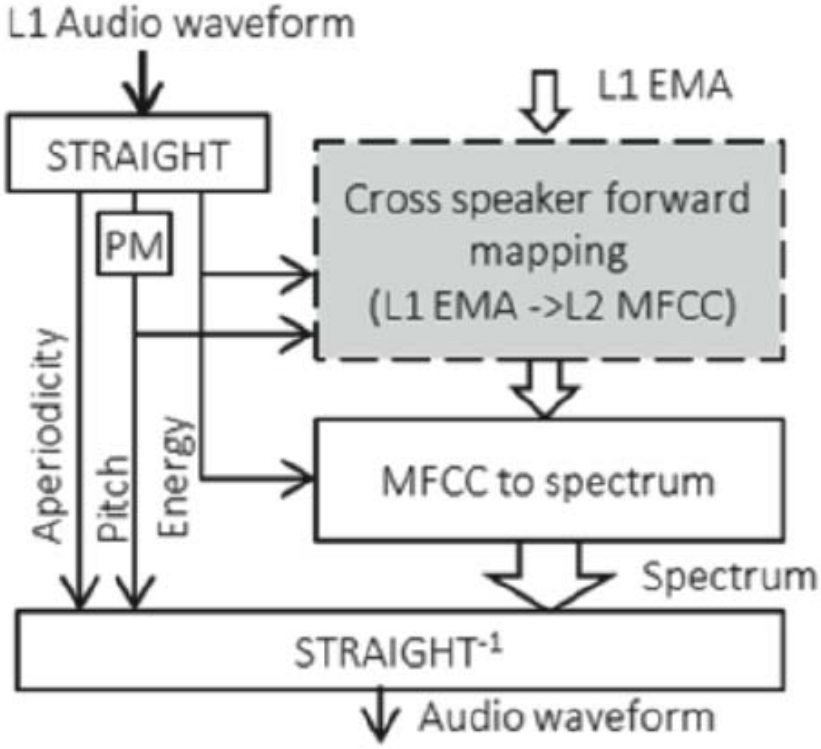
Insertion



Accent Conversion through articulatory synthesis



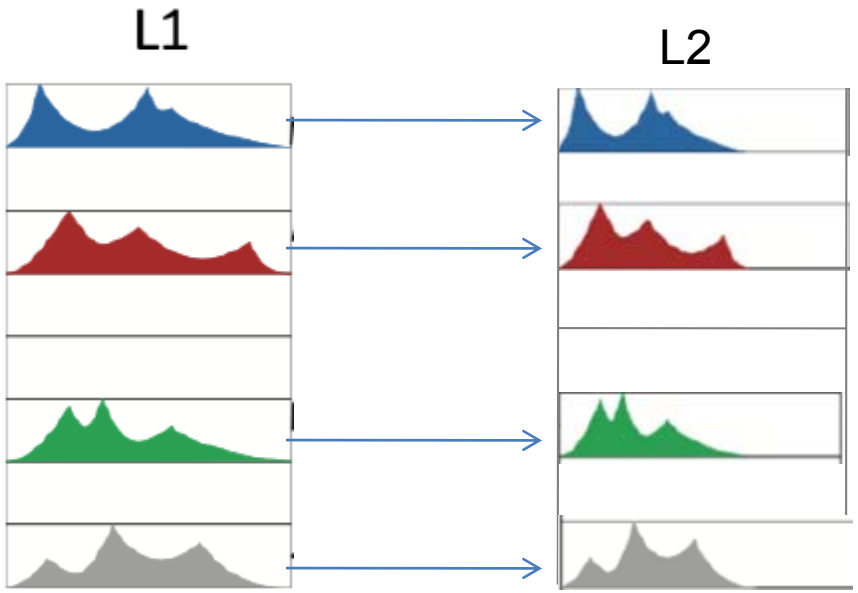
Baseline articulatory accent conversion system



Articulatory accent conversion with a cross speaker forward mapping

Electromagnetic articulography (EMA)

Source and target utterances are paired based on their wording in a forced aligned parallel corpus



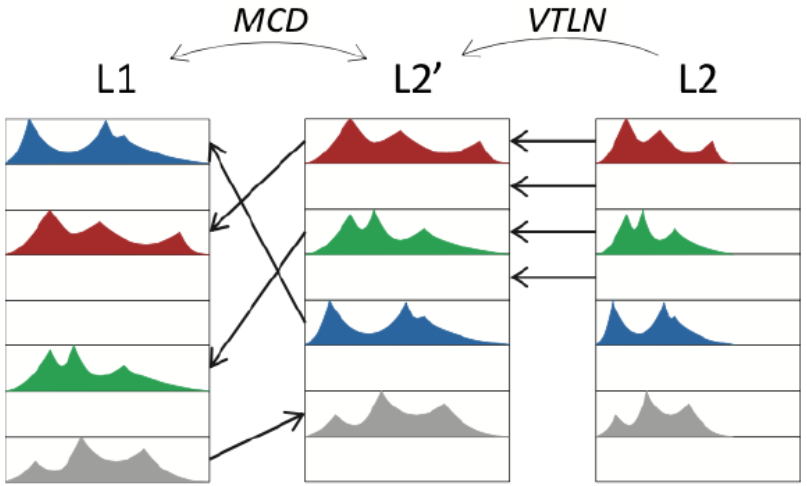
Accent Conversion through cross speaker articulatory synthesis

Step1. Apply VTLN to L2 acoustic vector $y^{(L2)}$ to account for physiological differences of both speakers' vocal tract.

$$\boldsymbol{W} = \arg\min \left\| \boldsymbol{y}^{(L1)} - \boldsymbol{W}.\boldsymbol{y}^{(L2)} \right\|^2$$

Step2. For each vector $y^{(L1)}$ we find its closet vector $y^{(L2)}$ * to minimize Mel Cepstral distortion (MCD) between them as

$$\boldsymbol{y}^{(L2)^*} = \arg\min_{\forall y_2} \left\| \boldsymbol{y}^{(L1)} - \boldsymbol{W}.\boldsymbol{y}^{(L2)} \right\|^2$$



repeat;

Step3. Discard pairs according to preselected threshold

Step4. Replace $y^{(L1)}$ to $x^{(L1)}$, getting look up table {$x^{(L1)}$, $y^{(L2)}$ * }

Original

Modified