CCS369- TEXT AND SPEECH ANALYSIS

UNIT IV

OVERVIEW

At a high level, voice applications have three main components: speech recognition, speech profiling, and speech synthesis.

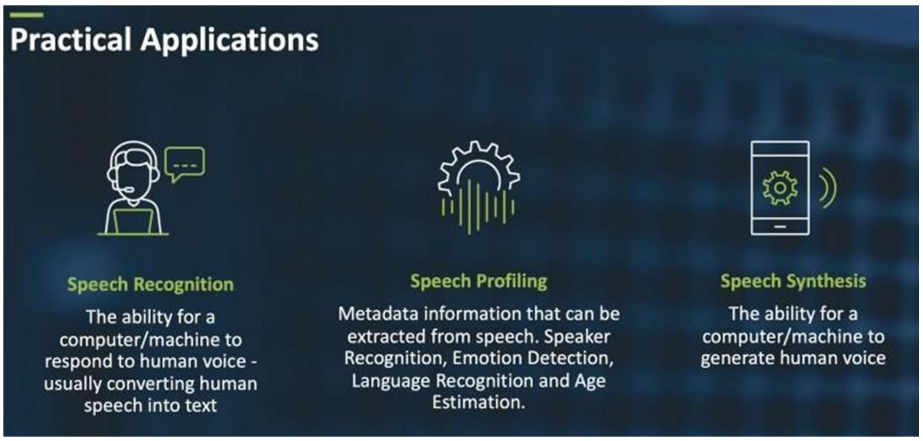


Figure 1: There are three main components of voice applications.

- Speech recognition is the translation of spoken language into text. It is also called automatic speech recognition, computer speech recognition, and speech to text. A major application of ASR is transcribing conversations. Other examples include "Hey, Siri" commands, such as "call home," and voice interfaces requiring simple data entry (e.g., entering credit card numbers or call routing—"press or say '1' for customer service").
- Speech profiling is the process of audio mining information from speech beyond recognition, including age, gender, emotion, the language spoken, speaker verification, etc. Applications include biometrics, sentiment analysis, and metadata extraction to improve customer intelligence initiatives.
- Speech synthesis is the artificial creation of human speech. In this post we'll occasionally use the term "speech synthesis" to refer to technologies that cut across TTS and speech synthesis. The more familiar term is "text-to-speech" but we're opting for "speech synthesis" because we expect input sources in the future to include a range of formats including text and audio. An example of a system that can take audio input is Tencent's PitchNet: a model that takes audio of one singing voice and converts it into audio of another voice singing the same content.

Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech. The reverse process is speech recognition. A TTS engine converts written text to a phonemic representation, then converts the phonemic representation to waveforms that can be output as sound.

Text-To-Speech Synthesis *is a machine learning task that involves converting written text into spoken words. The goal is to generate synthetic speech that sounds natural and resembles human speech as closely as possible.*

Use Cases

Text-to-Speech (TTS) models can be used in any speech-enabled application that requires converting text to speech imitating human voice.

Voice Assistants

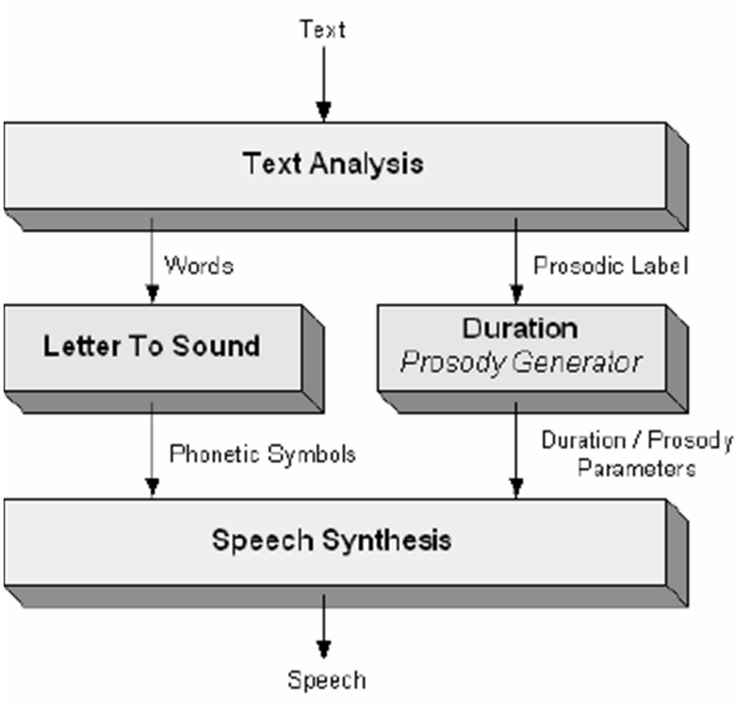
TTS models are used to create voice assistants on smart devices. These models are a better alternative compared to concatenative methods where the assistant is built by recording sounds and mapping them, since the outputs in TTS models contain elements in natural speech such as emphasis.

Announcement Systems

TTS models are widely used in airport and public transportation announcement systems to convert the announcement of a given text into speech

Some common applications:

- Conversational interactive voice response (IVR) systems, as in customer service call centers
- Voice commerce applications, such as shopping on an Amazon Alexa device
- Voice guidance and navigation tools, like GPS mapping apps
- Smart home devices and other voice-enabled Internet of Things (IoT) tools
- Independent virtual assistants like Apple's Siri, but for your own brand
- Experiential marketing and advertising solutions, like interactive voice ads on music streaming services or branded smart speaker apps
- Video game development, with dynamic runtime TTS for accessibility features, scene prototyping, and AI non-player characters
- Company training and marketing videos that allow creators to change voice-overs without tracking down original voice talent for ongoing recording sessions



TTS ARCHITECURE

A typical TTS system consists of two basic processing modules: **text analysis module and a speech synthesis module.**

Text analysis

The first module of a TTS pipeline, the text analysis module, is responsible for converting the incoming text into a linguistic representation that encodes the information of how the input text should be spoken.

As the first step, the module must process the text into a standardized format by taking care of any special characters or other inconsistencies. Then the text must be structurally analyzed to identify syntactical components of the sentences. Any inconsistencies between the written and spoken language (e.g., abbreviations and acronyms, proper names, numbers) must be detected and converted into a correct format (e.g., to map the string "100 sq ft" into "*one hundred square feet*", not "*one-zero-zero sq ft*") .

The second step consists of inferring the correct pronunciation of the words, also known as **grapheme-to-phoneme conversion.** Since the relationship between written and spoken language is not always straightforward, the text analysis module must convert strings of input letters into strings of phonemes based on the pronunciation conventions of the given language. This involves resolving many ambiguities prevalent in languages, such as how to deal with homographs, that is, words with the same written form but different pronunciations. Foreign language words and loan words have to be handled as well.

As the final stage, suprasegmental characteristics of the speech-to-be-produced must be introduced to the linguistic representation. Durations of the phonemes and intermediate pauses must be defined, even though the information does not exist in the text. In order to make the speech natural sounding and intelligible, the process also includes introduction of any potential rhythmic structure, stress patterns, and intonational cues to the linguistic representation. For this purpose, the text analysis module must interpret syntactic properties of the input text. For instance, the system must be able to differentiate different sentence types such as questions from statements and to infer which word should receive focus in the given sentential context.

Speech Synthesis

Speech synthesis systems aim at producing intelligible speech signals from some type of input language representation. Synthesis systems are also often referred to as text-to-speech (TTS) systems, as written text is a natural way to instruct what type of utterances should be produced. Speech Synthesis software are transforming the work culture of different industry sectors. A speech synthesizer is a computerized voice that turns a written text into a speech. It is an output where a computer reads out the word loud in a simulated voice; it is often called text-to-speech. It is not only to have machines talk simply but also to make a sound like humans of different ages and gender. With the rise of usage of digital services and the increase in dependency on voice recognition, the text-to-speech engine is gaining popularity.

There are 3 stages in which speech synthesis works; text to words, words to phonemes, and phonemes to sound.

Text to words –

The initial stage of speech synthesis, is generally called pre-processing or normalization, it is everything about reducing ambiguity: it's about narrowing down the many different ways a person could read a piece of text into the one that's the most appropriate. In Pre-processing it's about going through the text and then cleaning it up so the computer makes fewer mistakes when it reads the words aloud. Elements like numbers, dates, times, abbreviations, acronyms, and special characters need to be turned into words. This is however harder than it sounds. For example; the number 1953 might refer to several items, a year or a time, or a padlock combination; each of these is read out will sound slightly differently. While humans can figure out the pronunciation based on the way the text is written, computers generally don't have that ability to do that.

This is the reason they use statistical probability techniques or neural networks to arrive at the most likely pronunciation. If there were a decimal point before the numbers (".953"), then it would be read differently as "nine fifty-three." Pre-processing also handles homographs, these are the words pronounced in different ways but the meaning is different for each word. The word "sell" can be pronounced as "cell", so a sentence such as "I sell the flower" is problematic for a speech synthesizer. But if it can understand that the preceding text entirely has a different meaning, by recognizing the spelling ("I have a cell phone"), then it can make a reasonable guess that "I sell the pen" is likely correct.

Words to phonemes –

Once they figure out the words that need to be spoken, next the speech synthesizer has to generate the speech sounds that make up these words. Every computer needs is a huge alphabetical list of words and details of how to pronounce each word. For each word, they would need a list of the phonemes that make up its sound.

Theoretically, if a computer has a dictionary of words and phonemes, then all it needs to do is to read a word and look it up in the list, and then read out the corresponding phonemes. But practically, it's quite harder than it sounds.

The alternative approach involves breaking down the written words into their graphemes (written component units, typically made from the individual letters or syllables that make up a word) and then generate phonemes that correspond to them using a set of simple rules. The benefit of doing this is that the computer can make a reasonable attempt at reading any word.

Phonemes to sound –

At this point, the computer has converted the text into a list of phonemes. But how to find basic phonemes that the computer reads aloud when it's turning the text into speech. There are three different approaches to this.

- First to use recordings of humans saying the phonemes
- The second is for the computer to generate the phonemes itself by generating basic sound frequencies
- The last approach is to imitate the technique of the human voice.

Letter to Sound (LTS): Typically, there are **two possible solutions** to build a LTS module:

1) manually collect rules, with sufficient experience and language knowledge. This method is commonly referred to **rule-based method;**

2) acquire rules by machine learning, which is called **training-based method** generally.

Rule-based method of LTS Module

For a given language, if there exists a systematic relationship between a word format and its pronunciation, rule-based letter-to-sound can be efficient. During our research, we found Spanish, Italian, Portuguese and German are such kind of languages. The workflow of rule-based letter-to-sound is as follows. Firstly, we invite experienced linguists to create a set of pronunciation rules for a language. The programmer translates the rules that are expressed by natural language into a set of machine-readable rules. At the same time, we collect a number of pronunciation items for testing the rules. Secondly, the predicted items are compared with standard items to see what is the precision of the current rule set. If the precision is not high enough, we may modify the rule set and the codes. This interactive process is repeated until the precision reaches a predefined level within the acceptable code size.

Training-based method of LTS Module

The training-based method shows its advantages where there is no language knowledge available, or the work to write the pronunciation rule set systematically is too difficult. Many comparative experiments show that it can achieve higher accuracy than rule-based approach. The typical processes of training method include: *letter-to-phoneme alignment (or grapheme-to-phoneme alignment), model training, and testing.* Of course, for embedded applications, we always need to achieve tradeoff between model size and accuracy.

Letter-to-phoneme alignment

The letter-phoneme alignment task as the problem of inducing links between units that are related by pronunciation. Each link is an in- stance of a specific mapping between letters and phonemes.

TEXT NORMALIZATION

As part of a text-to-speech (TTS) system, the text normalization component is typically one of the first steps in the pipeline, converting raw text into a sequence of words, which can then be passed to later components of the system, including word pronunciation, prosody prediction, and ultimately waveform generation.

Text normalization converts text from written form into its verbalized form, and it is an essential preprocessing step before text-to-speech (TTS). It ensures that TTS can handle all input texts without skipping unknown symbols.

For example, "$123" is converted to "one hundred and twenty-three dollars."

Inverse text normalization is a part of the automatic speech recognition (ASR) post-processing pipeline. It converts ASR model output into its written form to improve text readability.

For example, the ITN module replaces "one hundred and twenty-three dollars" transcribed by an ASR model with "$123."

Inverse text normalization not only improves readability but also boosts the performance of downstream tasks such as neural machine translation or named entity recognition, as such tasks use written text during training.

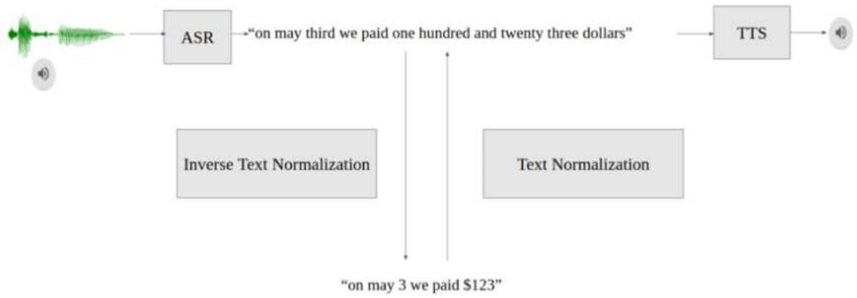


Figure 2: Text Normalization and Inverse Text Normalization

Text normalization and Inverse text normalization tasks face several challenges:

- Labeled data is scarce and difficult to collect.
- There is a low tolerance for unrecoverable errors, as normalization errors cascade down to subsequent models. Normalization errors that alter the input semantics are called unrecoverable.

Text normalization and Inverse text normalization systems support a wide variety of *semiotic classes*, that is, words or tokens where the spoken form differs from the written form, requiring normalization. Examples are dates, decimals, cardinals, measures, and so on.

Steps to Text Normalization

There are four main steps in text normalization:

- **Case normalization**
- *Tokenization and stop word removal*
- *Parts-of-Speech (POS) tagging*
- *Stemming.*

REFER UNIT 1 for the detailed explanation

Case normalization

Case normalization applies to languages that use uppercase and lowercase letters. All languages based on the Latin alphabet or the Cyrillic alphabet (Russian, Mongolian, and so on) use upper- and lowercase letters. Other languages that sometimes use this are Greek, Armenian, Cherokee, and Coptic. In the case normalization technique, all letters are converted to the same case. It is quite helpful in semantic use cases. However, in other cases, this may hinder performance. In the spam example, spam messages may have more words in all-caps compared to regular messages.

Case normalization provides case-normalized alternatives for words which, by their position in a sentence or because they occur in a title, may or may not appear with their inherent, meaningful capitalization.

Letter-to-sound

Letter to sound module plays a very important role in a TTS system. Letter-to-sound (LTS) conversion is a process used in linguistics, natural language processing, and speech synthesis to convert written text, typically in the form of letters or characters, into corresponding phonetic representations or sound sequences. The primary goal of letter-to-sound conversion is to enable text-to-speech (TTS) systems to produce spoken language from written text.The process of letter-to-sound conversion involves several key steps:

1. **Text Analysis:** The first step is to analyze the input text, typically at the level of individual words or sub-word units. This involves breaking down the text into its constituent letters or characters and determining how they are pronounced in the given language. Languages often have specific rules for how letters are pronounced in different contexts.

2. **Grapheme-to-Phoneme Rules:** Graphemes are the written representations of letters or characters, and phonemes are the basic units of sound in a language. Letter-to-sound conversion uses rules, often referred to as grapheme-to-phoneme rules, to map each grapheme to its corresponding phoneme. These rules can be language-specific and may account for various factors like letter combinations, word stress, and syllable structure.

3. **Contextual Considerations:** In many languages, the pronunciation of a letter or character can vary depending on the letters surrounding it. Letter-to-sound conversion systems may take into account the context of a letter within a word to generate the correct phonemic representation.

4. **Lexicon and Exception Handling:** Some words or names may not follow the regular grapheme-to-phoneme rules and require exceptions to be included in the system's lexicon. A lexicon is a dictionary that contains mappings of specific words or names to their corresponding phonetic representations.

5. **Prosody and Intonation:** In addition to converting individual letters to phonemes, letter-to-sound conversion may also consider aspects of prosody, such as word stress, intonation patterns, and pitch, to produce more natural-sounding speech.

Letter-to-sound conversion is crucial for the development of text-to-speech (TTS) systems, as it enables these systems to generate human-like speech from written text. TTS systems can be used in various applications, such as *voice assistants, audiobooks, accessibility tools for the visually impaired, and more*. Accurate letter-to-sound conversion is essential for the overall quality and naturalness of the synthesized speech.

It is difficult for a person to create a letter-to-sound module because it is hard to be a professional speaker of many languages. For a large TTS system, the orthodox method is to store a pronunciation lexicon with a reasonable number of items. And only apply letter to sound module to those not listed in this lexicon. But for embedded systems, it is not a smart way to use this method, because the lexicon will sharply increase the engine's size. The improved method is for the most frequently used words, apply letter-to-sound, for the words with high usage frequency but can NOT get the right prediction with a letter-to-sound module, add them to an exception list.

There are two types of Letter-to-sound modules:

- **Rule-based Letter-to-sound Module**
  This module manually collects rules, with sufficient experience and language knowledge. This method is commonly referred to as the rule-based method. For a given language, if there exists a systematic relationship between a word format and its pronunciation, rule-based letter-to-sound can be efficient. The workflow of rule-based letter-to-sound is as follows. Firstly, it invite experienced linguists to create a set of pronunciation rules for a language. The programmer translates the rules that are expressed by natural language into a set of machine-readable rules. At the same time, we collect a number of pronunciation items for testing the rules. Secondly, the predicted items are compared with standard items to see what is the precision of the current rule set. If the precision is not high enough, we can modify the rule set and the codes. This interactive process is repeated until the precision reaches a predefined level within the acceptable code size.
- **Training-based Letter-to-sound Module**
  This module acquires rules by machine learning, which is called a training-based method generally. A training-based letter-to-sound module learns letter-to-sound conversion rules and patterns from data during a training process. This module is designed to automatically generate phonetic representations (pronunciations) for words or text based on a dataset of text examples and their corresponding phonetic transcriptions.

PROSODY ANALYSIS

Prosody analysis is the examination and study of the acoustic, rhythmic, melodic, and intonational features of speech that go beyond the individual phonemes and words. It encompasses the analysis of suprasegmental elements, which are aspects of speech that span multiple phonemes or syllables. These elements are critical for understanding the rhythm, melody, and emotional content of spoken language. Prosody analysis involves the following key components:

1. **Pitch:** Pitch refers to the perceived highness or lowness of a speaker's voice. Prosody analysis involves measuring changes in pitch, known as intonation, which can indicate the rising or falling patterns in speech. Pitch variations are used to convey emphasis, question or statement intonation, and emotional expression.

2. **Duration:** Duration refers to the length of time that speech sounds, syllables, words, or phrases are produced. Prosody analysis considers variations in speech duration to determine the pacing, rhythm, and pauses in speech, all of which contribute to the overall expressiveness and meaning of spoken language.

3. **Intensity:** Intensity, also known as loudness, is a measure of the strength of the acoustic signal. Prosody analysis examines variations in intensity to determine how loud or soft specific parts of speech are. These variations can convey emotional expressiveness, emphasis, and sentence-level stress.

4. **Speech Rate:** Speech rate is the speed at which a speaker delivers speech. Prosody analysis involves measuring speech rate and tempo, which can indicate aspects like nervousness, excitement, and deliberate communication style.

5. **Rhythm and Timing:** Prosody analysis includes studying the rhythm of speech, which encompasses the regularity or irregularity of stressed and unstressed syllables. Rhythmic patterns in speech contribute to the cadence and flow of spoken language.

6. **Emotional Expression:** Prosody analysis is instrumental in identifying emotional content in speech. Variations in pitch, intensity, and rhythm can convey emotions such as happiness, sadness, anger, or surprise. This is particularly important in applications like sentiment analysis and emotion recognition.

7. **Pragmatic and Discourse Functions:** Prosody plays a crucial role in signaling pragmatic information in conversation, including turn-taking, distinguishing between questions and statements, conveying emphasis, and even indicating sarcasm.

Prosody analysis has a wide range of practical applications, including:

- **Speech Recognition:** It helps improve the accuracy of automatic speech recognition systems by considering prosodic cues in addition to phonetic content.

- **Text-to-Speech Synthesis:** In speech synthesis, prosody analysis is used to generate more natural and expressive synthesized speech by incorporating appropriate intonation, stress, and pacing.

- **Emotion Recognition:** Prosody analysis is a key component in recognizing emotional states from spoken language, which is valuable in applications such as virtual assistants and customer service.

- **Language Learning and Teaching:** Prosody analysis is used to teach and learn the prosodic features of a language, including its intonation patterns and speech rhythm.

- **Psycholinguistics and Cognitive Science:** It plays a role in studying how humans perceive and process prosody in speech, shedding light on the cognitive mechanisms involved in language understanding.

Overall, prosody analysis is essential for understanding the subtleties and nuances of spoken language, as it provides crucial cues beyond the mere content of words, contributing to effective communication, natural speech synthesis and recognition, and conveying emotional and pragmatic information.

PROSODIC EVALUATION

Prosody evaluation is the process of assessing and analyzing the prosodic features of speech. Prosody refers to the rhythm, melody, and intonation of speech, as well as other suprasegmental aspects, such as stress, pitch, tempo, and pausing. Prosody plays a crucial role in conveying the emotional and communicative aspects of spoken language and can greatly affect how a message is interpreted. Prosody evaluation aims to measure and understand these features for various purposes, including speech analysis, speech synthesis, language learning, and more. Here are some key aspects of prosody evaluation:

1. **Pitch and Intonation:** Pitch refers to the perceived frequency of a person's voice. Intonation involves the rise and fall of pitch in speech. Prosody evaluation assesses pitch patterns to determine how they contribute to the meaning and emotional expression in speech.

2. **Stress and Emphasis:** Prosody evaluation examines how speakers use stress and emphasis to highlight certain words or syllables, which can change the meaning or emphasis in a sentence.

3. **Tempo and Timing:** It assesses the tempo or speech rate, including factors like speech speed, speech rate variation, and timing of pauses. Prosody evaluation can reveal whether a speaker's tempo is appropriate for the context or if it conveys urgency, excitement, or other emotions.

4. **Rhythm and Syllable Structure:** The evaluation may also focus on the rhythm and syllable structure of speech, looking at how syllables are grouped and pronounced in connected speech.

5. **Emotional Expression:** Prosody plays a vital role in conveying emotions and attitudes in speech. Evaluation may assess how well a speaker conveys emotions such as happiness, sadness, anger, or sarcasm through their prosody.

6. **Speech Synthesis:** In the context of speech synthesis (text-to-speech systems), prosody evaluation is used to ensure that the generated speech sounds natural and conveys the intended meaning. This can involve evaluating the system's ability to produce varied intonation patterns, stress, and pacing.

7. **Language Learning and Teaching:** Prosody evaluation is used in language learning and teaching to help learners improve their pronunciation, stress patterns, and intonation in a foreign language. It can provide feedback to learners on how well they are reproducing the prosody of the target language.

8. **Speaker Identification:** Prosody evaluation can also be used in speaker identification or verification systems, where the unique prosodic features of a person's speech can serve as a biometric identifier.

Methods for prosody evaluation include perceptual evaluations, acoustic analysis, and machine learning techniques. Perceptual evaluation involves human listeners who rate or categorize speech based on prosodic features. Acoustic analysis uses software tools to extract prosodic parameters from speech recordings. Machine learning techniques can be employed to automatically analyze and classify prosodic features.

Overall, prosody evaluation is a multidisciplinary field that helps us better understand how the melody and rhythm of speech impact communication and can be applied in various domains to enhance the quality and effectiveness of spoken language.

SIGNAL PROCESSING

Signal processing plays a fundamental role in speech synthesis, which is the process of generating artificial speech from text or other forms of input. The main objective of signal processing in speech synthesis is to manipulate and transform digital representations of speech signals to produce natural-sounding and intelligible synthetic speech. Speech synthesis systems aim at producing intelligible speech signals from some type of input language representation. Synthesis systems are also often referred to as text-to-speech (TTS) systems, as written text is a natural way to instruct what type of utterances should be produced.

Synthesis algorithms

The second key module consists of the speech synthesizer module. Input to the synthesizer is the linguistic representation produced by the text analysis block while the output consists of acoustic waveforms of synthesized speech. There are several potential technical approaches to the creation of a speech waveform from linguistic instructions. Historically, methods such as formant synthesis or articulatory synthesis have been utilized (where the latter is still used in speech research). However, modern commercial speech synthesizers are based on one of the two alternative techniques: **concatenative synthesis** **or** **statistical parametric speech synthesis.** Both methods are described in more detail in their respective sub-sections.

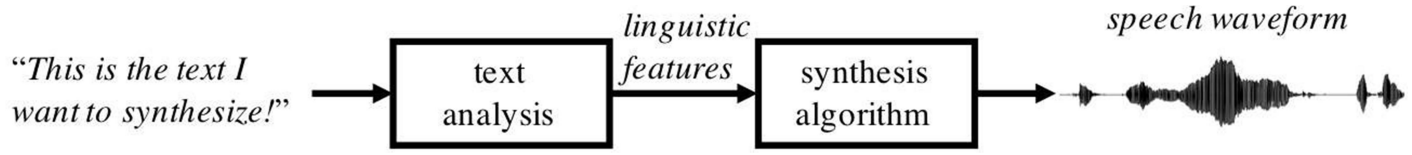


Figure 1: The basic structure of a speech synthesis system.

In general, synthesis algorithms aim at speech output that maximally resembles natural speech, is free of noise and artifacts, and has high intelligibility. Other characteristics may include possibility to use different speaker voices or to speaking styles to account for different use contexts and user preferences. In practical use, computational complexity of the system may also become a relevant design factor. This is especially true if the system must support real-time speech production and/or serve multiple users simultaneously. For instance, speech synthesis used on a standard mobile device or in a car entertainment system must operate with strict latency computational complexity constraints.

Speech quality and intelligibility a speech synthesizer is typically evaluated using subjective listening tests.

Concatenative synthesis is a technique for synthesising sounds by concatenating short samples of recorded sound (called units). The duration of the units is not strictly defined and may vary according to the implementation, roughly in the range of 10 milliseconds up to 1 second.

Statistical parametric synthesis might be most simply described as generating the average of some set of similarly sounding speech segments. This contrasts directly with the desire in unit selection to keep the natural unmodi- fied speech units, but using parametric models offers other benefits.

Concatenation synthesis

Concatenative synthesis is based on the concatenation (stringing together) of segments of recorded speech. Generally, concatenative synthesis produces the most natural-sounding synthesized speech. However, differences between natural variations in speech and the nature of the automated techniques for segmenting the waveforms sometimes result in audible glitches in the output. There are three main sub-types of concatenative synthesis.

Unit selection synthesis

Unit selection synthesis uses large databases of recorded speech. During database creation, each recorded utterance is segmented into some or all of the following: individual phones, diphones, half-phones, syllables, morphemes, words, phrases, and sentences. Typically, the division into segments is done using a specially modified speech recognizer set to a "forced alignment" mode with some manual correction afterward, using visual representations such as the waveform and spectrogram. An index of the units in the speech database is then created based on the segmentation and acoustic parameters like the fundamental frequency (pitch), duration, position in the syllable, and neighboring phones. At run time, the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This process is typically achieved using a specially weighted decision tree.

Unit selection provides the greatest naturalness because it applies only a small amount of digital signal processing (DSP) to the recorded speech. DSP often makes recorded speech sound less natural, although some systems use a small amount of signal processing at the point of concatenation to smooth the waveform. The output from the best unit-selection systems is often indistinguishable from real human voices, especially in contexts for which the TTS system has been tuned. However, maximum naturalness typically requires unit-selection speech databases to be very large, in some systems ranging into the gigabytes of recorded data, representing dozens of hours of speech. Also, unit selection algorithms have been known to select segments from a place that results in less than ideal synthesis (e.g. minor words become unclear) even when a better choice exists in the database. Recently, researchers have proposed various automated methods to detect unnatural segments in unit-selection speech synthesis systems.

Diphone synthesis

Diphone synthesis uses a minimal speech database containing all the diphones (sound-to-sound transitions) occurring in a language. The number of diphones depends on the phonotactics of the language: for example, Spanish has about 800

diphones, and German about 2500. In diphone synthesis, only one example of each diphone is contained in the speech database. At runtime, the target prosody of a sentence is superimposed on these minimal units by means of digital signal processing technique or more recent techniques such as pitch modification in the source domain using discrete cosine transform. Diphone synthesis suffers from the sonic glitches of concatenative synthesis and the robotic-sounding nature of formant synthesis, and has few of the advantages of either approach other than small size. As such, its use in commercial applications is declining, although it continues to be used in research because there are a number of freely available software implementations.

Domain-specific synthesis

Domain-specific synthesis concatenates prerecorded words and phrases to create complete utterances. It is used in applications where the variety of texts the system will output is limited to a particular domain, like transit schedule announcements or weather reports. The technology is very simple to implement, and has been in commercial use for a long time, in devices like talking clocks and calculators. The level of naturalness of these systems can be very high because the variety of sentence types is limited, and they closely match the prosody and intonation of the original recordings.

Because these systems are limited by the words and phrases in their databases, they are not general-purpose and can only synthesize the combinations of words and phrases with which they have been preprogrammed. The blending of words within naturally spoken language however can still cause problems unless the many variations are taken into account. For example, in non-rhotic dialects of English the *"r"* in words like *"clear"* /ˈklɪə/ is usually only pronounced when the following word has a vowel as its first letter (e.g. *"clear out"* is realized as /ˌklɪəɹˈʌʊt/). Likewise in French, many final consonants become no longer silent if followed by a word that begins with a vowel, an effect called liaison. This alternation cannot be reproduced by a simple word-concatenation system, which would require additional complexity to be context-sensitive.

Statistical Parametric Speech Synthesis (SPSS)

A complete SPSS system is generally composed of three modules: a text analysis module, a parameter prediction module which uses a statistical model to predict the acoustic feature parameters such as fundamental frequency (F0), spectral parameters and duration, and a speech synthesis module. The text analysis module mainly preprocesses the input text and transforms it into linguistic features used by the speech synthesis system, including text normalization , automatic word segmentation, and grapheme-to-phoneme conversion. These linguistic features usually include phoneme, syllable, word, phrase and sentence-level features. The purpose of the parameter prediction module is to predict the acoustic feature parameters of the target speech according to the output of the text analysis module. The speech synthesis module generates the waveform of the target speech according to the output of the parameter prediction module by using a particular synthesis algorithm. The SPSS is usually divided into two phases: the training phase and the synthesis phase. In the training phase, acoustic feature parameters such as F0 and spectral parameters are firstly extracted from the corpus, and then a statistical acoustic model is trained based on the linguistic features of the text analysis module as well as the extracted acoustic feature parameters. In the synthesis phase, the acoustic feature parameters are predicted using the trained acoustic model with the guidance of the linguistic features. Finally, the speech is synthesized based on the predicted acoustic feature parameters using a vocoder.

Formant synthesis

Formant synthesis does not use human speech samples at runtime. Instead, the synthesized speech output is created using additive synthesis and an acoustic model (physical modelling synthesis). Parameters such as fundamental frequency, voicing, and noise levels are varied over time to create a waveform of artificial speech. This method is sometimes called *rules-based synthesis*; however, many concatenative systems also have rules-based components. Many systems based on formant synthesis technology generate artificial, robotic-sounding speech that would never be mistaken for human speech. However, maximum naturalness is not always the goal of a speech synthesis system, and formant synthesis systems have advantages over concatenative systems. Formant-synthesized speech can be reliably intelligible, even at very high speeds, avoiding the acoustic glitches that commonly plague concatenative systems. High-speed synthesized speech is used by the visually impaired to quickly navigate computers using a screen reader. Formant synthesizers are usually smaller programs than concatenative systems because they do not have a database of speech samples. They can therefore be used in embedded systems, where memory and microprocessor power are especially limited. Because formant-based systems

have complete control of all aspects of the output speech, a wide variety of prosodies and intonations can be output, conveying not just questions and statements, but a variety of emotions and tones of voice.

Articulatory synthesis

Articulatory synthesis refers to computational techniques for synthesizing speech based on models of the human vocal tract and the articulation processes occurring there. The first articulatory synthesizer regularly used for laboratory experiments was developed at Haskins Laboratories in the mid-1970s by Philip Rubin, Tom Baer, and Paul Mermelstein. This synthesizer, known as ASY, was based on vocal tract models developed at Bell Laboratories in the 1960s and 1970s by Paul Mermelstein, Cecil Coker, and colleagues.

Until recently, articulatory synthesis models have not been incorporated into commercial speech synthesis systems. A notable exception is the NeXT-based system originally developed and marketed by Trillium Sound Research, a spin-off company of the University of Calgary, where much of the original research was conducted. Following the demise of the various incarnations of NeXT (started by Steve Jobs in the late 1980s and merged with Apple Computer in 1997), the Trillium software was published under the GNU General Public License, with work continuing as gnuspeech. The system, first marketed in 1994, provides full articulatory-based text-to-speech conversion using a waveguide or transmission-line analog of the human oral and nasal tracts controlled by Carré's "distinctive region model".

More recent synthesizers, developed by Jorge C. Lucero and colleagues, incorporate models of vocal fold biomechanics, glottal aerodynamics and acoustic wave propagation in the bronchi, trachea, nasal and oral cavities, and thus constitute full systems of physics-based speech simulation.

HMM-based synthesis

HMM-based synthesis is a synthesis method based on hidden Markov models, also called Statistical Parametric Synthesis. In this system, the frequency spectrum (vocal tract), fundamental frequency (voice source), and duration (prosody) of speech are modeled simultaneously by HMMs. Speech waveforms are generated from HMMs themselves based on the maximum likelihood criterion.

Sinewave synthesis

Sinewave synthesis is a technique for synthesizing speech by replacing the formants (main bands of energy) with pure tone whistles.

Deep learning-based synthesis

Deep learning speech synthesis uses Deep Neural Networks (DNN) to produce artificial speech from text (text-to-speech) or spectrum (vocoder). The deep neural networks are trained using a large amount of recorded speech and, in the case of a text-to-speech system, the associated labels and/or input text. It is known that the HMM-based speech synthesis method maps linguistic features into probability densities of speech parameters with various decision trees. Different from the HMM-based method, the DL-based method directly perform mapping from linguistic features to acoustic features with deep neural networks which have proven extraordinarily efficient at learning inherent features of data. In the long tradition of studies that adopt DL-based method for speech synthesis, people have proposed numerous models. a brief overview of the advantages and disadvantages in Table 1 and makes a detailed introduction in the following.

Table 1. The advantages and disadvantages of different speech synthesis methods, including hidden Markov model (HMM), restrictive Boltzmann machine (RBM), deep belief network (DBN), deep mixture density network (DMDN), deep bidirectional long short-term memory (DBLSTM), WaveNet, Tacotron and convolutional neural network (CNN).

Methods	Advantages	Disadvantages
HMM	Flexible with changing voice characteristics and the system is robust	The acoustic features are oversmoothed, making the generated speech sounds muffled
RBM	Can better describe the distribution of high-dimensional spectral envelopes to alleviate the over-smooth problem	Suffer from the fragementation problem of training data
DBN	Cannot suffer from the training data fragementation problem and reduce the over-smoothing problem	The quality of generated speech will be degraded
DMDN	Can solve the single modality problem	Can only leverage limited contexts and each frame is mapped independently
DBLSTM	Can fully leverage contextual information	Still needs a vocoder to synthesize waveform
WaveNet	Can produce high-quality speech waveforms	Too slow and the errors from the front-end will affect the synthesis effect
Tacotron	Fully end-to-end speech synthesis model and can produce high-quality speech waveforms	Quite costly to train the model
CNN	Fast to train the model	The speech quality might be degraded

4.1. Restrictive Boltzmann Machines for Speech Synthesis

In the field of speech synthesis, Boltzmann machine (RBM) is usually regarded as a density model for generating the spectral envelope of acoustic parameters. It is adopted to better describe the distribution of high-dimensional spectral envelopes to alleviate the over-smooth problem in HMM-based speech synthesis.

4.2. Multi-distribution Deep Belief Networks for Speech Synthesis

Multi-distribution deep belief network (DBN) [24] is a method of modeling the joint distribution of context information and acoustic features. It models the coutinuous spectral, discrete voiced/unvoiced (V/UV) parameters and the multi-space F0 simultaneously with three types of RBMs. In DBNs, the visible unit can obey different probability distributions, therefore, it is possible to characterize the supervectors that are composed of these features.

4.3. Deep Bidirectional LSTM-based Speech Synthesis

BLSTM-RNN is an extended architecture of bidirectional recurrent neural network (BRNN). It replaces units in the hidden layers of BRNN with LSTM memory blocks. With these memory blocks, BLSTM can store information for long and short time lags, and leverage relevant contextual dependencies from both forward and backward directions for machine learning tasks.

When using deep BLSTM-based (DBLSTM) model to predict acoustic parameters for speech synthesis, first we need to convert the input text prompt into a feature vector, and then use the DBLSTM model to map the input feature to acoustic parameters. Finally, the parameter generation algorithm is used to generate the acoustic parameters and a vocoder is utilized to synthesize the corresponding speech.

4.4. End-to-End Speech Synthesis

A TTS system typically consists of a text analysis front-end, an acoustic model and a speech synthesizer. Since these components are trained independently and rely on extensive domain expertise which are laborious, errors from each component may compound. To address these problems, end-to-end speech synthesis methods which combine those components into a unified framework have become the main stream of speech synthesis field. In the following we will give a brief introduction to the end-to-end speech synthesis methods.

4.4.1. Speech Synthesis Based on WaveNet

WaveNet is a complete probabilistic autoregressive model that predicts the probability distribution of the current audio sample based on all samples which have been generated before. As an important component of WaveNet, dilated causal convolutions are used to ensure that WaveNet can only use the sampling points from 0 to t - 1 while generating the tth sampling point. The original WaveNet model uses autoregressive connections to synthesize waveforms one sample at a time, with each new sample conditioned on the previous ones.

4.4.2. Speech Synthesis Based on Tacotron

Tacotron is a fully end-to-end speech synthesis model. It is capable of training speech synthesis model given <text, audio> pairs, thus alleviating the need for laborious feature engineering. And since it is based on character level, it can be applied in almost all kinds of languages including Chinese Mandarin. Tacotron uses seq2seq model with attention mechanism to map text to spectrogram which is a good representation of speech. Since spectrogram doesn't contain phase information, the system uses Griffin-Lim algorithm to reconstruct the audio by estimating the phase information from the spectrogram iteratively.