Automatic Speech Recognition (ASR)

How ASR works and how Voicebox overcomes real-world challenges to ASR.

ASR is the first stage in an overall human/computer interaction pipeline that also includes Voicebox’s related Natural Language Understanding (NLU) and Text-to-Speech (TTS) technologies. Voicebox’s advanced ASR module is a multi-stage pipeline that uses techniques from machine learning, graph theory, traditional grammar development, and statistical analysis of large corpuses to form high-confidence transcriptions of input audio.

Real-world concerns—everything from accent and dialect differences, to CPU and memory limitations on different platforms, to operation despite environmental noise, and more—present additional challenges beyond what academic ASR research can achieve in controlled laboratory environments. Voicebox’s ASR system uses a wide range of techniques to overcome those obstacles and maintain robust performance under real-world conditions.

Voicebox offers ASR solutions for a wide range of CPU, memory, and connectivity limitations. The Voicebox ASR can support fully-embedded operation on small-footprint devices, hybrid combinations of embedded and server-based ASR, and fully connected scenarios in which the ASR runs on a high-performance server in the cloud.
What is ASR?

Automated speech recognition is the process of using software to convert a person’s spoken words into a text transcription.

In past decades, the job of transcribing text belonged to people. Stenography was a mainstream career path for women entering the workforce in the 1960s, 70s, and 80s. In the 1990s, as computing power grew, computers began taking over this work in limited contexts. Early speech recognition applications were largely focused on dictation—creating documents by voice. Mobile computing was in its infancy, and only desktop-class machines could perform ASR well enough.

Today, advances in mobile computing have put the power of ASR into billions of cell phones, automotive consoles, and Internet of Things (IoT) devices worldwide. The limitations of these platforms—notably, their lack of physical keyboards—has created a vast need for an easy, accurate, natural form of interaction with those devices.

Using voice and ordinary language has been the dream of human/computer interaction since the idea was first introduced to a mass audience through the 1966 television series Star Trek.

Real advances in computer technology have far outstripped even the vision of Star Trek in terms of miniaturization, communications, and the overall ubiquity of computation. Yet, due to the practical challenges of speech processing, the dream of voice interaction has remained of science fiction for over half a century.

But today, as the bridge between the user and Voicebox’s related technologies of Natural Language Understanding (NLU) and Text-to-Speech (TTS), ASR brings that dream to reality.
Inside the Voicebox ASR Module

The Voicebox ASR module is a multi-stage pipeline. Each stage converts the input audio signal into a more abstract form which can ultimately be mapped onto words. The pipeline’s six stages proceed as follows.

1. The input audio signal is broken up into “frames” containing a small amount of audio data each. Typically, frames hold between 10 and 16 milliseconds of audio. The ASR performs a spectral analysis and from that extracts acoustic vectors of 40 components that represent the acoustic properties of each frame.

2. The sequence of acoustic frames is the input to a deep neural network stage. Voicebox trains the neural network on a large corpus of audio samples with known transcriptions. Voicebox has built this corpus over many years in order to fine-tune ASR performance to the specific domains and vocabularies needed by the speech systems we have produced. The neural network’s output is a sequence of probabilities that the input audio contains particular abstract acoustic states. These acoustic states are the building blocks of phonemes.

3. The ASR then applies the DNN’s output probabilities to an acoustic states graph representing the transitions between acoustic states. The probabilities indicate the DNN’s confidence that a given transition actually happened, and therefore that the corresponding acoustic states were present sequentially within the input audio.

4. The ASR then performs a Viterbi search through the acoustic states graph in order to find sequences of acoustic states which have a high joint probability of occurring, and which correspond to known acoustic state sequences of various phonemes in the target language.

5. The Viterbi search yields a graph of potential phoneme sequences. The transitions between phonemes also have probabilities, which are a function of the target language’s statistics (i.e. what sound sequences the language uses most and least) and the joint probabilities from the acoustic states graph.

6. The ASR performs another Viterbi search, this time against the phoneme graph, to find high probability sequences of phonemes which correspond to words in the ASR’s word graph. The word graph is built jointly from the pre-compiled grammars that specify what utterances the ASR should recognize, plus n-gram probabilities extracted from a large corpus of text in the target language. A final search through the word graph identifies the highest-probability paths, which indicate the ASR’s best hypothesis for the input audio signal’s transcription.

The ASR’s processing pipeline is illustrated on the following page.
Audio Signal

Acoustic Vectors

Deep Neural Network

Probabilities

Acoustic States Graph

Phoneme Graph

Word Graph

ASR Output

[“how”, “do”, “i”, “get”, “to”, “Seattle”]
ASR in the Real World

There is a substantial difference between laboratory, proof-of-concept ASR systems and real-world ASR systems. Laboratory systems can work around many problems and completely ignore others. Voicebox’s ASR module has no such luxury. This section introduces the problems facing the ASR module and how Voicebox solves them to maintain robust performance under real-world conditions.

Vocabulary

As discussed above, the Word Graph stage in the ASR’s processing pipeline is built from grammars which describe the set of utterances the ASR should recognize. These may be domain-specific grammars corresponding to specific command-and-control scenarios, or grammars for general-purpose text entry.

In either situation, the ASR will be confronted with vocabulary that was not in the grammar. For example, a command-and-control grammar for telephony should be expected to recognize utterances such as “Call Alan on his cell phone.” While the ASR can be trained on a corpus that includes many common given names, users will inevitably also ask to call people with uncommon names or names that are foreign relative to the ASR’s target language.

Voicebox uses two different strategies to overcome this issue.

The first method is limited in its ability to immediately recognize out-of-scope words, but provides immediate results. In this method, the Viterbi search algorithm on the phoneme graph can be relaxed to output high-probability phoneme sequences even if they don’t correspond to known words. That is, if the ASR has high confidence that the phoneme sequence /lakʃmi/ occurred in the input, it can output “lakshmi” (a given name) as a likely spelling by mapping the phonemes to their common representations in the target language.

The second method, called “continuous improvement,” enables the ASR to recognize anything, but does not provide immediate results. In this method, the ASR’s output is monitored for unrecognized words. Any such words can be manually transcribed and added to the system’s training data. This method is very useful on server-based ASR systems running in the cloud, where the overall system maintenance schedule provides opportunities for periodically updating the ASR’s word graph.
Accents and Dialects

Most languages include groups of speakers who have different accents or who speak in different dialects of the language. Accents and dialects affect the sequence of phonemes a given user produces, which in turn affects the ASR’s ability to match that sequence to its word graph.

Voicebox overcomes this data by training the ASR against data from a wide range of speakers, with a wide range of backgrounds. Voicebox actively collects sample recordings from speakers with different backgrounds, including non-native speakers. This data enables the ASR’s deep neural network (stage 2 in the ASR Processing Pipeline diagram) to correctly learn the acoustic properties of different accents and dialects, and enables the phoneme graph (stage 5) to recognize variant pronunciations of particular words.

Noise and Far-Field ASR

In ideal laboratory conditions, an ASR’s input can be recorded on high-quality microphones and in a quiet environment, resulting in a very high-quality input that contains nothing but the user’s speech. In the real world, people use ASRs in noisy environments, microphones are often low-quality condenser microphones integrated into cell phones or other devices, and far-field situations where the microphone is at some distance from the user.

These factors lead to audio input with a lower signal-to-noise ratio that is more difficult to accurately process. Voicebox uses different strategies to overcome each of these issues.

Acoustic Modeling

ASR systems can remove some background noise from the input signal using signal processing techniques. These can provide limited improvement in targeted scenarios (e.g. removing typical road noise from audio recorded inside a moving vehicle) but tend not to work well in the general case. Voicebox’s ASR has adopted a different noise-rejection strategy, called acoustic modeling.

An acoustic model is the set of data which informs the ASR’s deep neural network how the different phonemes of speech sound under a particular set of environmental conditions. Voicebox’s corpus of audio recordings covers many different audio environments. By training the DNN against multiple acoustic models, the system learns to hear those phonemes as-is, even with background noise. This strategy generalizes much better than earlier signal-processing methods, without requiring additional CPU resources for signal processing.
Far-Field Techniques

The term “far-field” refers to situations in which the ASR’s microphone is relatively far from the speaker. Far-field situations are common in home appliance scenarios where the appliance is situated somewhere in the same room as the user, but not necessarily close to them.

The main challenges in far-field ASR are reverberation and echoes. Reverberation is when the microphone records slightly time-delayed copies of the user’s speech signal, as that signal bounces off of walls and other surfaces in the surrounding environment. Echos are similar, but refer to removing a known audio signal—for example, whatever song the system is currently playing—from the microphone input so as not to contaminate the user’s speech signal.

Ideally, the microphone would only record sound from the direct audio path, while rejecting sound from reverberation and echo paths. As this is not possible without very specialized recording setups, audio filtering techniques apply to these far-field noise sources. The two main techniques are called beam-forming and echo cancellation.

Beam forming requires that the system have multiple microphones that are arranged in a known geometry. Often, several small microphones spaced evenly a few centimeters apart, forming an array microphone. Audio signals arrive at each microphone at slightly different times, owing to each microphone’s different placement within the array. These time delays change depending on the direction of the incoming sound. If the individual signals are added together with appropriate offsets, the combined signal forms a “beam” aimed in a specific direction—such as the direction towards the user. This beam amplifies sound within the beam while suppressing sound from other directions. Adaptive beam forming adjusts the offsets to steer the beam towards the user, and are appropriate when the user’s position is unpredictable.
or variable. Static beam-forming solutions are appropriate when the user’s location is predictable, such as recording the driver of a car.

Echo cancellation works by estimating a “transfer function” between a known signal coming out of the system’s speakers, and the signal recorded by the system’s microphone(s). This transfer function models how the surrounding environment returns a modified echo of the speaker output back to the microphone. This transfer function is applied to the known audio signal, resulting in an estimated noise signal that is then subtracted from the microphone’s output. This allows the system to maintain good ASR performance even when the user is talking over other output sounds.

Audio Path Engineering

Finally, the physical and electronic components that deliver the user’s speech to the ASR also affect the ASR’s performance. Microphone quality obviously matters, but so do the properties of the op-amps, analog-to-digital chips, and other components involved in converting the raw audio into digital samples the ASR can consume. Poorly chosen components can degrade the input in a variety of ways, such as low bit-depths or even line-level interference from other components in the device.

Voicebox has well over a decade of experience in working with hardware manufacturers to diagnose these issues in their prototype systems, so as to recommend design changes that maximize signal-to-noise ratio over the full audio path while respecting parts costs.

Latency

Processing voice input requires greater computing resources than traditional keyboard/mouse/touchscreen input. While modern hardware can process voice input, the computational load still translates into a noticeable lag between when a user finishes speaking and when their device responds to what they said.

Minimizing this latency is a key issue in maintaining a positive user experience.

Voicebox’s “Streaming ASR” technology is the principle method for minimizing latency. From the user’s perspective, latency only matters from the moment the user finishes speaking until the system responds. However, most voice commands take three to five seconds for the user to say, or up to ten seconds for complex commands.

The user’s device is largely idle while the user is speaking. Older voice systems required the device to record the user’s entire utterance before sending it to the ASR. Voicebox’s Streaming
ASR makes use of the user’s speaking time by starting the ASR process as soon as the user begins speaking, so the ASR can run in parallel with the user's speaking time.

On modern hardware, the ASR can perform approximately in real-time, meaning that a given duration of speech takes about that same amount of time to process. Thus, under streaming ASR, by the time the user is done speaking the ASR is very nearly done transcribing their speech. The result is a much more responsive voice interface.

Resource Limitations

Voice is the ideal way of interacting with embedded and mobile devices. And yet, those devices are the ones least likely to have powerful CPUs and large amounts of memory. Fortunately, Voicebox has developed strategies for bringing ASR to all platforms, regardless of their resource limitations.

Grammar Scalability

The memory requirements of an ASR system scale according to the size of the ASR’s vocabulary and the number of different utterances the ASR needs to recognize. The first strategy for enabling ASR on resource-limited devices is to carefully optimize the ASR’s vocabulary and grammar so that it fits within a target device’s memory.

Dynamic Grammars

Mobile and embedded devices may lack the ample memory space of server-class hardware, but modern devices often have considerable non-volatile storage. Another strategy on these devices is to use multiple, smaller grammar files which are dynamically loaded as the ASR needs to recognize different things. Using dynamic grammars and carefully optimizing what material is covered by each one, Voicebox can achieve very strong ASR coverage even on memory-constrained devices.

Off-board and Hybrid ASR

Increasingly reliable network connectivity opens other avenues for ASR on limited devices.

Devices with extremely limited resources can offload ASR processing to a server in the cloud. The device streams audio to the server, which then returns the transcription to the device. Server-class hardware can accommodate very large vocabularies and grammars. Servers can also serve many clients at once and may be more economical than adding resources to the device. Server round-trips also add some latency to each voice request, although to some extent this is balanced by the high performance of server-class hardware. A greater concern can be
making ASR dependent on network connectivity; devices with mission-critical requirements for ASR functionality can run a limited ASR locally to handle critical commands.

Mid-range devices can combine these strategies in what is called “Hybrid ASR.” In this model, the device uses on-board ASR with a limited vocabulary and grammar for low-latency recognition of essential voice commands, plus off-board ASR for higher-latency recognition of a wide range of utterances. An arbitration module on the device compares the results from each ASR to decide which one to use.

Voicebox’s ASR Products

Voicebox offers ASR options that scale to any level of computing resources, from embedded devices up to cloud-based solutions.

For small-scale embedded platforms, we offer the Sensory ASR. This ASR is well-suited for applications that use keyword-activation to trigger speech functionality, and only need to support a small number of commands and command variations (generally less than 1000). Sensory is a kilobyte-scale ASR in terms of the size of speech recognition grammar it supports.

For mid-range platforms, Voicebox offers the Talisman ASR engine. Talisman supports megabyte-scale ASR grammars, requires PC-class or modern smartphone-class hardware, and supports vocabulary sizes and command variations in the hundreds of thousands.

At the highest end, the Voicebox Cloud Platform offers a server based ASR that supports gigabyte-scale grammars. These are able to recognize extremely large vocabularies, such as catalogues of millions of song titles, as well as millions of command variations, all with excellent low-latency performance.

Next Steps

Learn more about Voicebox Technologies and our ASR, Conversational Voice AI, and Text-to-Speech solutions at voicebox.com.

Sign up for the Voicebox Developer Portal, at developer.voicebox.com, to try our cloud services APIs and speech-enable your own applications.

Sign up for our monthly mailing list by e-mailing newsletter@voicebox.com.