Microsoft

Transparency Note: Speech-to-Text

Updated 11/12/2021

Table of Contents

What is a Transparency Note?

An AI system includes not only the technology, but also the people who will use it, the people who will be affected by it, and the environment in which it is deployed. Creating a system that is fit for its intended purpose requires an understanding of how the technology works, what its capabilities and limitations are, and how to achieve the best performance. Microsoft's Transparency Notes are intended to help you understand how our AI technology works, the choices system owners can make that influence system performance and behavior, and the importance of thinking about the whole system, including the technology, the people, and the environment. You can use Transparency Notes when developing or deploying your own system, or share them with the people who will use or be affected by your system.

Microsoft's Transparency Notes are part of a broader effort at Microsoft to put our AI Principles into practice. To find out more, see the Microsoft AI principles.

Introduction to Speech-to-Text

Speech-to-Text, also known as automatic speech recognition (ASR), is a feature under Speech Services (See What is Speech-to-Text?), that converts spoken audio into text. Speech-to-Text supports over 140 locales for inputs. Please see the latest list of supported locales in Language and voice support for the Speech service.

Terms and definitions

Term	Definition
Audio input	The streamed audio data or audio file that is used as an input for the speech-to-text feature. Audio input may contain not only voice, but also silence and non-speech noise. Speech-to-Text generates text for the voice parts of audio input.
Utterance	A component of audio input that contains human voice. One utterance may consist of a single word or multiple words, such as a phrase.
Transcription	The text output of the Speech-to-Text feature. This automatically generated text output leverages speech models (defined below) and is sometimes referred to as machine transcription or automated speech recognition (ASR). Transcription in this context is fully automated and therefore different from human transcription, which is text generated by human transcribers.
Speech model	An automatically generated, machine-learned numerical representation of an utterance used to infer a transcription from an audio input. Speech models are trained on voice data that includes various speech styles, languages, accents, dialects and intonations, as well as acoustic variations generated by different types of recording devices. A speech model numerically represents both acoustic and linguistic features, which are used to predict what text should be associated with the utterance.
Real-time API	An API that accepts requests with audio input, and returns a response in real time with transcription within the same network connection.
Language Detection API	A type of real-time API that detects what language is spoken in a given audio input. A language is inferred based on voice sound in the audio input.
Speech Translation API	Another type of real-time API that generates transcriptions of a given audio input then translates them into a language specified by the user. This is a cascaded service of Speech Services and Text Translator.

Term	Definition
Batch API	A service to send audio input to be transcribed at a later time. Customers specify location(s) of audio file(s) with other parameters, such as a language name. The service loads the audio input asynchronously and transcribes it. Once transcription is complete, text file(s) are loaded back to a location specified by the customer.
Speaker Separation (diarization)	Speaker Separation (also called diarization) answers the question of who spoke when. It differentiates speakers in an audio input based on their voice characteristics. The Batch API supports diarization and is capable of differentiating two speakers' voices on mono channel recordings. Diarization is combined with speech to text functionality in order to provide transcription outputs that contain a speaker entry for each transcribed phrase. This diarization feature only supports the differentiation of two voices, and the transcription output is tagged as Speaker 1 or 2. If more than two speakers are within the audio, this will result in inaccurate tagging.

Speech-to-Text features

There are two ways to use the Speech-to-Text service - Real-time and Batch.

Real-time Speech-to-Text API

This is a common API call via Speech SDK/REST API to send an audio input and receive a text transcription in real time. The speech system uses a "speech model" to recognize what is spoken in a given input audio. A speech model consists of an acoustic model and a language model, which combine to calculate likely spoken content. The acoustic model maps a small linguistic unit called a "phoneme" to voice sound in an audio input, which leads to a probability of specific word or phrase. The language model calculates probabilities of word combinations in a specified language. The Speech-to-Text technology then infers a final phrase or text based on combined probabilities. This feature is often used, for example, for voice-enabled queries or dictation within a customer's service or application.

Sub-features and options of the Real-time Speech-To-text API

- **Language Detection:** Unlike in a default API call, where a language (or a locale) for an audio input must be specified in advance, with language detection, customers can specify multiple locales and let the service detect a language.
- **Speech Translation:** This API converts audio input to text, and then translates and transcribes it into another language. The translated transcription output can be returned in text format or the customer may choose to have the text synthesized into audible speech by Text-to-Speech (TTS). See What is the Translator service for more information about the text translation service.

Batch transcription API

This is another type of API call, typically used to send lengthy audio inputs and to receive transcribed text asynchronously (i.e., at a later time). To use this API, users can specify locations of multiple audio files. The Speech-to-Text technology reads the audio input from the file streams and generates transcription text files which are returned to the customer's specified storage location. This feature is used to support larger

transcription jobs (e.g., an audio broadcast for later text transcription) where it is not necessary to provide end users the transcription content in real time.

Sub-features and options of the Batch Transcription API

- **Speaker Separation (diarization):** This feature is disabled by default. If a customer chooses to enable this feature, the service will differentiate between up to two speakers' utterances. The resulting transcription text contains a "speaker property" that indicates either Speaker 1 or Speaker 2, which denotes which of two speakers is speaking in an audio file. Speaker separation only works with two speakers. If audio input sent via the Batch API contains more than two speakers, the service will still separate voice into two speakers, leading to errors in speaker tags tied to transcription output.

Example use cases

Speech-to-Text can offer more convenient or alternative ways for end users to interact with applications and devices. Instead of typing words on a keyboard or using their hands for touch screen interactions, Speech-to-Text technology allows users to operate applications and devices by voice and through dictation.

- **Smart Assistants** -- Companies developing smart assistants on appliances, cars, and homes can use Speech-to-Text to enable natural interface search queries or command-and-control features by voice.
- **Chat Bots** -- Companies can build chat bot applications, in which users can use voice enabled queries or commands to interact with bots.
- **Voice Typing** -- Apps may allow users to use voice to dictate long-form text. Voice typing can be used to enter text for texting, emails, and documents.
- **Voice Commanding** -- Users can trigger certain actions by voice. This is called "command and control", and typical examples are entering query text by voice and selecting a menu item by voice.
- **Voice translation** -- Customers can use speech translation features of Speech-to-Text technology to communicate by voice with other users who speak different languages. Speech translation enables voice-to-voice communication across multiple languages. See the latest list of supported locales in Language and voice support for the Speech service.
- **Call Center transcriptions** -- Customers often record conversations with their end users in scenarios such as customer support calls. Audio recordings can be sent to the Batch API for transcription.
- **Mixed-language dictation** -- Customers can use Speech-to-Text technology to dictate in multiple languages. Using Language Detection, a dictation application may automatically detect spoken languages and transcribe appropriately without forcing users to specify which language they speak.

Considerations when choosing other use cases

Speech-to-Text API can offer users convenient options for enabling voice enabled applications, but it is very important to consider the context within which you will integrate the API and ensure that you comply with all laws and regulations that apply to your application. This includes understanding your obligations under privacy and communication laws, including national and regional eavesdropping and wiretap laws, that apply to your jurisdiction. Only collect and process audio that is within the reasonable expectations of your users; this includes ensuring you have all necessary and appropriate permissions from users for your collection, processing and storage of audio data.

Many applications are designed and intended to be used by a specific individual user for voice enabled queries, commands or dictation, however, the microphone for your application may pick up sound or voice from non-primary users. To avoid capturing the voices of non-primary users unintentionally, you should take the following into consideration.

Microphone considerations: Often you cannot control who may speak around the input device that sends audio input to the Speech-to-Text cloud service. You should therefore encourage your end-users to take extra care when using voice enabled features and applications in a public/open environment where other people's voices may be easily captured and/or for allowing non-primary users to use the voice-enabled application.

Only use Speech-to-Text in experiences and features that are within the reasonable expectations of your users: Audio data of a person speaking is personal information. Speech-to-Text is not intended to be used for covert audio surveillance purposes, in a manner that violates legal requirements, or in applications and devices in public spaces or locations where users may have a reasonable expectation of privacy. Use speech services only to collect and process audio in ways that are within the reasonable expectations of your users; this includes ensuring you have all necessary and appropriate permissions from users for your collection, processing, and storage of audio data.

Characteristics and limitations of Speech-to-Text

Speech-to-Text recognizes what's spoken in an audio input and generates transcription outputs. This requires proper setup for an expected language used in the audio input and spoken styles. Non-optimal settings may lead to lower accuracy.

Language of accuracy

The industry standard to measure Speech-to-Text accuracy is Word Error Rate (WER). WER counts the number of incorrect words identified during recognition, then divides by the total number of words provided in correct transcript (often created by human labeling).

To understand the detailed WER calculation, please see this document about Speech-to-Text evaluation and improvement.

Transcription Accuracy and System Limitations

Speech-to-Text uses a unified speech recognition machine learning model to understand what is spoken in a wide range of contexts and topic domains, such as command-and-control, or dictation and conversations. Therefore, you don't need to consider using different models for your application or feature scenarios.

However, you need to specify a language (or locale) for each audio input. This must match with actual language spoken in an input voice. Please check the list of supported locales for more details.

There are many factors that lead to a lower accuracy in transcription.

- **Acoustic quality:** Speech-to-Text enabled applications and devices may use a wide variety of microphone types and specifications. The unified speech models have been created based on various voice audio device scenarios, such as telephones, mobile phones and speaker devices.

However, voice quality may be degraded by the way users speak to microphones, even if they use high-quality microphones. For example, if users stay far from a microphone, the input quality would be too low, while speaking too close to a microphone would cause audio quality deterioration. Both cases may adversely impact Speech-to-Text accuracy.

- **Non-speech noise:** If an input audio contains a certain level of noise, it has an impact on accuracy. Noise may come from the audio devices used to make a recording, while audio input itself may contain noise, such as background noise and environmental noise.
- **Overlapped speech:** There could be multiple speakers within range of an audio input device, and they may speak at the same time. Also, other speakers may speak in the background while the main user is speaking.
- **Vocabularies:** The Speech-to-Text model has knowledge of wide variety of words in many domains. However, users may speak company specific terms and jargon (which are "out of vocabulary"). If a word appears in the audio that doesn't exist in a model, it will result in a mis-transcription.
- **Accents:** Even within one locale (such as "English -- United States"), many people have different accents. Very particular accents may also lead to a mis-transcription.
- **Mismatched locales:** Users may not speak the languages you expect. If you specified English -- United States (en-US) for an audio input, but a user spoke Swedish, for instance, accuracy would be worse.

Due to these acoustic and linguistic variations, customers should expect a certain level of inaccuracy in the output text when designing an application.

Best practices for improving accuracy

As discussed above, acoustic conditions, such as background noise, side speech, distance to microphone, speaking styles and characteristics can adversely affect the accuracy of what is being recognized.

Please consider the following application/service design principles for better speech experiences.

- **Design UIs to match input locales:** Mismatched locales will deteriorate accuracy. The Speech SDK supports automatic language detection, but it only detects one out of four locales specified at runtime. You still need to understand in what locale your users will speak. Your user interfaces should clearly indicate which languages the users may speak by such measures as a drop-down list of languages allowed to be used. Please see supported locales.
- **Allow users to try again:** Misrecognition may occur due to some temporary issues, such as unclear or fast speech or a long pause. If your application expects specific transcriptions, such as predefined action commands (such as "Yes" and "No"), and did not get any of them, users should be able to try again. A typical method is to tell users "Sorry, I didn't get that. Please try again".
- **Confirm before taking an action by voice:** Just as with keyboard/click/tap-based user interfaces, if an audio input can trigger an action, users should be given an opportunity to confirm the action, especially by displaying or playing back what was recognized/transcribed. A typical example is sending a text message by voice. An app repeats what was recognized and asks for confirmation: "You said, 'Thank you'. Send it or change it?".
- **Add custom vocabularies:** The general speech recognition model provided by Text-to-Speech covers a broad vocabulary. However, scenario specific jargon and named entities (people names, product names, etc.) may be underrepresented and what words and phrases are likely to be spoken can vary significantly depending on the scenario. If you can anticipate which words and phrases will be spoken (for instance, when a user selects an item from a list), you may want to

use the phrasal list grammar, see "Improving recognition accuracy" in Get started with Speech-to-Text.

- **Use Custom Speech:** If Speech-to-Text accuracy in your application scenarios remains low, you might want to consider customizing the model for your acoustic and linguistic variations. You may create your own models by training with your own voice audio data or text data. See more details about Custom Speech.

Fairness

At Microsoft, we strive to empower every person on the planet to achieve more. An essential part of this goal is working to create technologies and products that are fair and inclusive. Fairness is a multi-dimensional, sociotechnical topic and impacts many different aspects of our product development. You can learn more about Microsoft's approach to fairness here.

One dimension we need to consider is how well the system performs for different groups of people. Research has shown that without conscious effort focused on improving performance for all groups, it is often possible for the performance of an AI system to vary across groups based on factors such as race, ethnicity, region, gender and age.

Speech-to-Text models are trained and tuned with voice audio with variations including:

- Microphones and device specifications
- Speech environment
- Speech scenarios
- Languages and accents
- Age and genders
- Ethnic background

Each version of the Speech-to-Text model is tested and evaluated against various test sets to make sure the model can perform without a large gap in each of the evaluation criteria.

Evaluating Text-to-Speech in your applications

The performance of Text-to-Speech will vary depending on the real-world uses and conditions that customers implement. In order to ensure optimal performance in their scenarios, customers should conduct their own evaluations of the solutions they implement using Text-to-Speech.

A test voice dataset should consist of actual voice inputs collected in your applications in production. Customers should randomly sample data to reflect real user variations over a certain period of time. Additionally, the test dataset should be refreshed periodically to reflect changes in the variations.

Learn more about responsible AI

Microsoft AI principles

Microsoft responsible AI resources

Microsoft Azure Learning courses on responsible AI

Learn more about Speech-to-Text

What is Speech-to-text?

Contact us

Give us feedback on this document

About this document

Published: 11/12/2021

Last updated: 11/12/2021