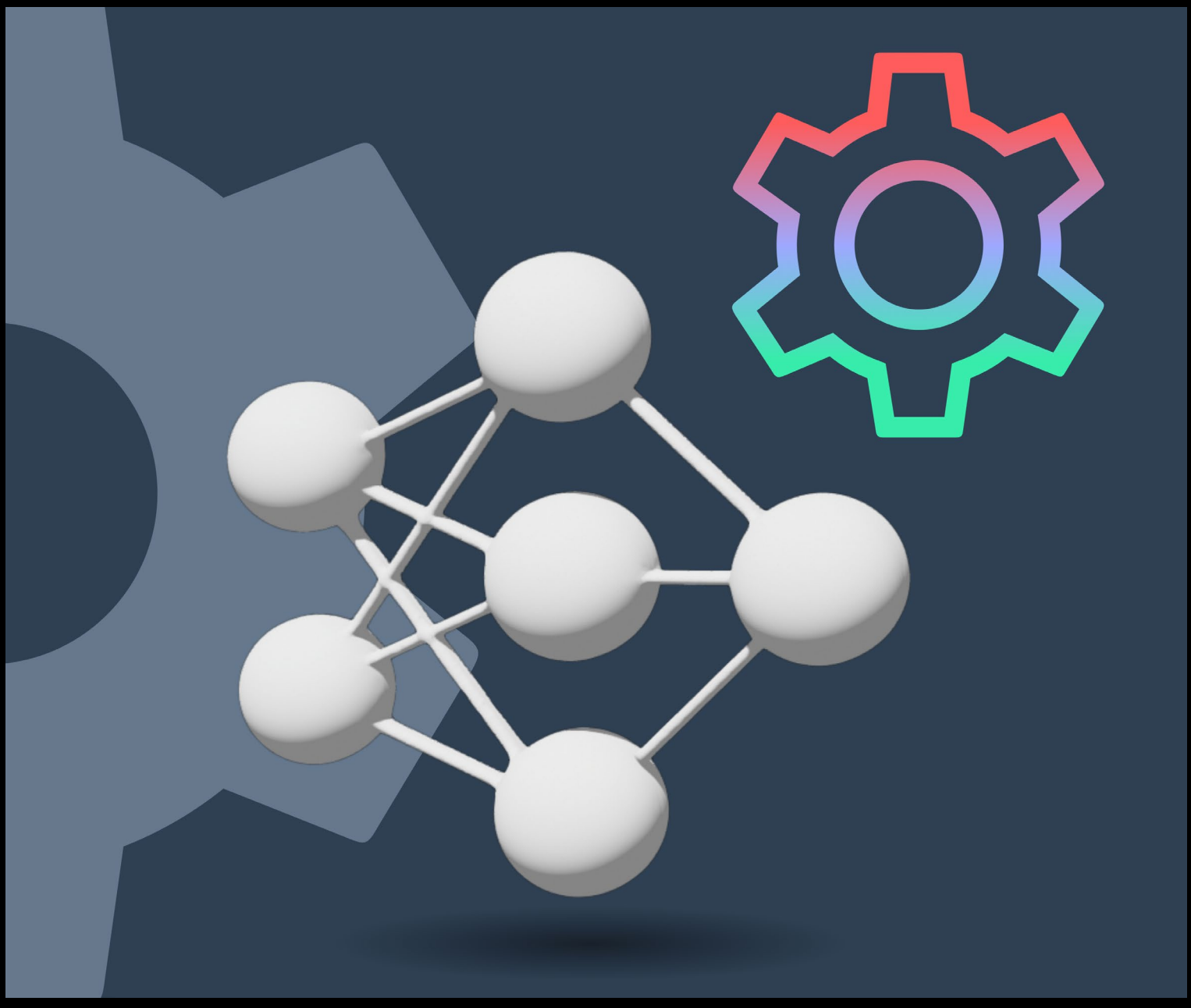
WHITE PAPER

How AI Speech Models Work



Contents

1 Executive Summary

What is a speech model and why should you care how it works? You might think that as long as a model is accurate enough for your purposes, you don't need to worry about what's happening behind the scenes. But that's not completely true. While accuracy is obviously critical for your business, using a speech model that has a fast and flexible architecture will pay short-term and long-term dividends above and beyond accuracy.

And for automatic speech recognition (ASR) solutions, the speech model is where the magic happens—it's where accuracy is created, speed is improved, and cost is determined. After building a great speech model, what's next? Do you spend time improving that one speech model to understand all dialects, accents, and terminology—or should you build multiple speech models for different dialects and use cases? The accuracy data shows that global, all-in-one language models can typically only reach about 75% accuracy. If you have any accents, dialects, jargon, acronyms, and industry terminology—i.e., the audio of most conversations, especially in a business setting—the accuracy decreases dramatically.

The leading edge of the speech industry is quickly shifting toward solutions that are flexible and customizable, able to meet the constantly-shifting demands of complex use cases and customers with diverse speech characteristics. And with the growing importance of data-centric AI, previously insurmountable problems are coming to the fore and being solved by those using cutting-edge technologies to build their speech models. In the near future, speech processing systems will be able to dynamically select the highest-accuracy speech model for an audio sample, ensuring the highest possible accuracy.

Not all ASR systems currently on the market are flexible and efficient enough to meet the emerging demands of the market. To reach human-level accuracy in speech recognition, the companies will need to adopt flexible end-to-end deep learning (E2EDL) architectures to enable transfer learning, rapid training on real-world audio data, and customization to meet the demands of diverse use cases, dialects, industry terminology, and jargon. This is why it's so critical that you go beyond accuracy in understanding and evaluating any ASR solution you might be considering.

This white paper explores the two main types of speech recognition models currently on the market, their architectures, and the characteristics that should be considered when deciding which speech model is right for your use case. We'll wrap up by talking about the impact that using the more modern model can have for your business beyond accuracy, and what the future holds for speech recognition.

2 Three Key Characteristics Influencing Speech Model Performance

Regardless of which type of ASR system you're looking at, these three key characteristics—**model design and architecture, training data**, and **model training**—are the foundation of any automatic speech recognition (ASR) or speech-to-text (STT) solution. This means that the accuracy, speed, scalability, compute resource, and costs are all affected by the design of the speech model. When you use any ASR solution, you're feeding your audio into one of these models; thus, the characteristics of the model will influence the results that you see.

Let's take a look at each of these three key characteristics of speech recognition systems in turn to understand how they work and how they contribute to what the final model is capable of.

1. Design and Architecture

A speech-to-text system is a set of computer algorithms and processes for creating properly formatted text from audio.

Today, there are two main types of speech models being used in speech recognition: legacy systems and end-to-end deep learning models. **Legacy systems** use a waterfall approach to speech recognition, in which several sub-processes that denoise audio, recognize phonemes, build words, and organize them into grammar are linked together in a sequence (see Figure 1 for an overview).

The more modern option is a speech model that uses **end-to-end deep learning** (E2EDL). This type of system is a comprehensive model that does all four of the things described above directly from the source audio without sequential sub-processing steps. E2EDL systems are faster, more computationally efficient, and more flexible, making them the ideal choice for enterprises today.

Let's take a close look at how each of these two systems works in order to better understand why E2EDL is the best approach available.

Legacy Speech Systems

The legacy technology used by most established speech companies uses a series of steps to go from audio input to readable transcript. This sequential process first denoices the audio, then analyzes acoustic qualities in an audio sample to generate phoneme predictions, then looks for word candidates, and finally assesses those candidates according to the probability of their occurring together in a grammatically-correct sentence.

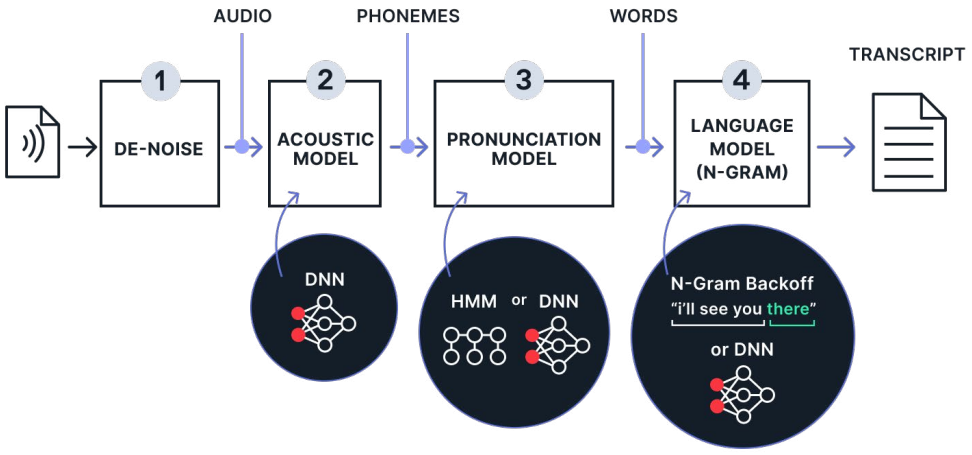


Figure 1. A schematic representation of a legacy speech model, showing the four main steps: denoising, an acoustic model, a pronunciation model, and a language model, as well as the machine learning alternatives that are available at each step.

These legacy systems use some combination of older approaches, like hidden Markov models (HMM) or *n*-gram backoff models (discussed in more detail below) alongside deep neural networks (DNN) at each stage of the process. The key difference between this kind of legacy system and an end-to-end deep learning one is that the use of machine learning in these legacy systems is limited and localized within each of the sequential steps. For example, a legacy speech model may use DNNs locally in the acoustic part of the overall speech model, but other, older tools for other parts of the process. It's the stringing together of different systems for different tasks that makes this legacy system hard to maintain and improve.

To better understand how these legacy systems operate, let's walk through an example of how such a system would process a bit of audio input.

After a denoising step (Step 1 in Figure 1), an acoustic model (Step 2) that has been trained on audio features identifies phonemes—the smallest meaningful units of sound in a language. The output of this process is a series of candidate speech sounds broken down into their constituent phonemes. For example, if the audio input was someone saying "speech sounds," the model would output: s - p - ee - ch s - ou - n - d - z. The more audio training data fed into the DNN, the better the DNN performs in determining the correct phonemes.

In a legacy system, these phonemes are then fed into the pronunciation model (Step 3). This is commonly a hidden Markov model (HMM), although this step can also use a second DNN. The pronunciation model will take the phonemes from Step 2 and determine the most probable word or words for a given sequence of phonemes.

Once candidate words are identified, there is yet another handoff to the last step in the sequence: the language model (Step 4). This is commonly a backoff n-gram model, but a DNN model or a combination of the two approaches are also possible.

An *n*-gram backoff model analyzes the probability of a word based on the previous *n* minus 1 words. For example, a 2-gram or bigram model estimates the current word based on the previous word. Taking our example above, if the model has determined that the first word is "speech", then it will be more likely to predict "sounds" as the next word, as those two words occur together more often than other, similar-sounding options like "speech pounds."

However, if the language model was a 4-gram model, then the probability of "speech" would increase further if the preceding words were, for example, "he produced," while other, acoustically similar options like "speed" would have a lower probability. The higher the *n* in the *n*-gram model or the better the training data for that speech use case the better the prediction. The downside is that the larger the *n*-gram, the more computing power the model requires.

The Problems with Legacy ASR Systems

All legacy speech systems share three common, fundamental flaws, regardless of which steps a given legacy system has replaced with DNNs.

1. **Each sub-model must be trained, optimized, updated, and maintained separately**. This not only creates a lot of day-to-day work—and thus cost—it also makes it challenging if you want to expand to new languages or use cases by updating the system, as each step will need to be updated independently and then rigorously tested to ensure that everything works correctly.
2. Because legacy systems only go in one direction—taking an input without any insight into what the steps before did—any issues with previous steps can cascade into future steps in a process called **error propagation**, which can negatively impact the accuracy of the system.
3. Related to the above, **the legacy approach is inherently lossy**, meaning that it generates a huge number of candidate guesses at each step—phonemes, words, word sequences—that it discards, as it passes on only its top candidates to the next step. This makes these systems computationally inefficient to run, driving up costs and slowing down processing speeds, while also decreasing the overall accuracy of the system. In other words, these legacy systems are not compute efficient and for any on-premise deployments, your compute cost will skyrocket as you audio volume goes up. Many legacy system require one CPU and one GPU per audio stream being processed.

End-to-End Deep Learning Speech Model

The alternative to the legacy approach is to use a single end-to-end deep neural network (DNN), a type of deep learning architecture that feeds input data through 10s to 1000s of processing layers. These systems process information in ways that mimic neurons in the human brain and produce algorithms that excel at highly complex tasks like speech processing. A schematic representation of how such a system works is shown in Figure 2.

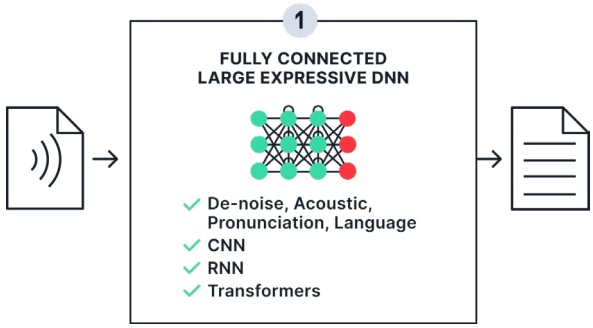


Figure 2. A schematic representation of an end-to-end deep learning (E2EDL) model.

Although DNNs can be used for some or all of the steps in the legacy system, the difference in an end-to-end deep learning (E2EDL) system is that there is only one step: a single model takes in audio and converts it to text. There are no separate acoustic, pronunciation, and language models with distinct inputs and probabilistic outputs.

An E2EDL model is created by feeding training audio data and human-generated transcripts into a DNN over many training cycles until the DNN develops a mature algorithm for recreating the results provided by the human transcriptionists. This process is described in the "Model Training" section below.

Overcoming the Problems of Legacy Systems

Because there is only a single model involved in E2EDL systems, they don't suffer from the same problems as legacy systems.

First, unlike legacy systems where each step must be trained and updated manually, an E2EDL system can be updated all at once, simply by changing the data that it's trained on. Depending on the needs of the system, an E2EDL speech model can be designed with a combination of convoluted neural networks (CNN), recurrent neural networks (RNN), and transformers. But regardless of what's "inside" the E2EDL model, the entire model is trained together, so that it can learn and improve all at once, rather than having to update different pieces of a larger system. This unified AI model training means faster accuracy improvements for the end user. You can correct for noise, train for jargon and terminology, and include accents in one step. Accuracy improvement can be seen in a few weeks instead of months.

These models can thus continue to improve as they are trained on more data, and so benefit from a continuous data flywheel to drive ongoing improvements. They are also highly useful for **transfer learning**, a process by which the algorithm from a mature model can be used as the starting point to train a new model quickly and easily. The new model can then undergo specialized training to rapidly create a custom model that meets the requirements of a specific customer or use case. Bottom line, you can customize a speech model for your specific audio characteristics in weeks using the real-world labeled audio data. Think of creating custom speech models for different dialects, different end customer product names, and and even different language mixtures like Spanglish and Hinglish.

Second, because there is only one step, there is no concern about error propagation or data loss. Each decision the model makes is based on the entirety of the data available, leading to better decisions and better accuracy as well. What's more, through targeted training, E2EDL models can learn to automatically output text, punctuation, and other kinds of speech analysis across a limitless variety of accents, dialects, and industry use cases—all that's needed is new data of the type that the model should predict on.

As for speed, multi-step legacy designs are normally slower and require more computing power as compared to an E2EDL design. Legacy designs may require both CPUs and GPUs, while an E2EDL design only requires GPUs, making scale-up easier. In addition, the better optimization of the speech model design, the lower the computational cost. For E2EDL AI speech models, you can run multiple transcription streams on one GPU. Notably, E2EDL models are easier to optimize and redesign to minimize costs, due to only having one system that needs to be updated, rather than three interlocking ones. E2EDL speech models can be up to 300 times faster than legacy models.

Now that we have an understanding of the two options for ASR systems on the market today, let's dive into the second key feature you need to know to understand speech models: the training data used to create the systems.

2. Training Data

Another important characteristic to consider when evaluating speech models is how they use training data. DNNs in either type of speech model can only pattern-match on data that they have been exposed to. Suppose you only trained your DNN on U.S. English; it would have a difficult time with a Scottish accent, like in this, our favorite comedy skit on speech recognition. It also might have accuracy issues with the accent of someone from Boston or the South if it has never been trained on the audio data for those dialects.

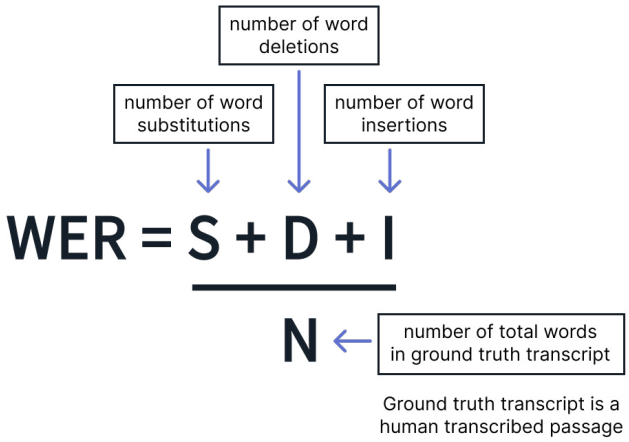
There are public datasets out there that can be used to get started in training your model, but it's important to understand the limitations of these datasets. Two example datasets are:

- ASPiRE contains approximately 226 hours of English speech with transcripts of interviews, transcript readings, and conversational telephone speech in the Philadelphia area.
- LibriSpeech is a large-scale corpus of around 1000 hours of English speech reading audiobooks.

Using this training data—which is specific to either speakers in noisy environments in Philadelphia or to audiobooks, respectively—with an open-source Kaldi speech model, you get the following results if the trained models are fed real-world audio of interviews and sales cold calls.

Data Used	Interviews Word Error Rate	Sales Cold Call Word Error Rates
ASPiRE	41%	33%
LibriSpeech	40%	26%

Industry-standard Word Error Rate (WER) is the measure of accuracy and is defined as:



The lower the WER, the higher accuracy and readability of the final transcript. For context, easy-to-read transcripts should have a WER of under 20%, and ideally closer to 10%, and trained human transcriptionists can reach WERs of 4-5%, allowing for some inevitable errors from background noise, crosstalk, slang, and other issues.

As you can see for audio interviews, neither dataset did very well, and the transcripts were unreadable. LibriSpeech was almost readable for sales cold calls.

The non-DNN components of the Legacy model would also need data to train on, and here the breadth of that data matters. Suppose you only had 100 words in the data dictionary. In that case, every predicted word would be one of those 100 words. As you expand the word dictionary, you will have more words to choose from, but the more data you add, the more computing power it takes to make accurate predictions.

Data makes the model. The more diverse the training data, the more accurately the model can predict diverse words, accents, dialects, jargon, and terms. This is now called data-centric AI. Audio data is curated and selectively used to create more accurate models for different use cases. Some speech models may be intentionally biased toward a specific use case, accent, language, or dialect. For example, Google and Amazon's STT are biased toward command-and-response applications, as that was where their STT started: i.e., "Alexa, turn on the lights" or "Ok Google, what is the weather today."

This type of bias can be good or bad depending on whether or not the technical bias works for your application. Any speech model can be made to be biased by specifically training and adding the right data. For real-world applications, the best speech models are ones that are fed the real-world audio data for that specific application or use case, creating a "bias" that actually makes them perform better for their specific scenarios.

3. Model Training

The last important characteristic of speech models is how they are trained. One point to clarify, model training is not keyword or key phrase boosting or using custom vocabulary. These methods do not change the speech model but only push the model to output certain words more frequently than similar words. AI speech model training teaches the AI model how to interpret the audio to create the correct output.

For legacy systems, each DNN is trained separately, which is time consuming, labor intensive, and expensive. If you are using three separate DNNs in your legacy system, then each DNN would be trained with a different set of data. Next, you have to optimize all the DNNs to create the best output.

For a single end-to-end DNN speech model, you are only training on one set of data. You can improve the entire process by providing one great set of training data. No optimization of other DNNs are required. This is a more efficient and faster process to improve accuracy. Now, let's look at the three general methods for model training.

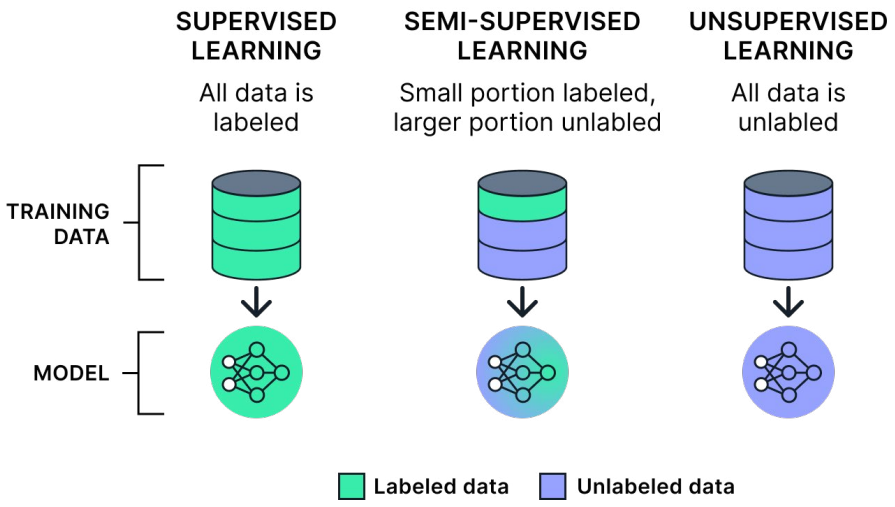


Figure 3. The three types of AI model training.

- **Supervised learning** is when you provide your model with labeled training datasets. Labeled training datasets comprise both the audio and the ground truth transcript—i.e., human-transcribed text. This tells your model what features of the audio are important to learn.
- **Semi-supervised learning** is a combination of supervised and unsupervised training. One approach is to perform pseudo-labeling where your model generates the labels for your unlabeled training data—e.g., this phone call has a negative, neutral, or positive sentiment.

This is helpful when you have a vast amount of data to label. Then this labeling is validated by human transcriptionists.

- **Self-supervised or unsupervised learning** is when you provide your model with audio files. The model then "extracts" information from the audio or clusters the data into groups. This is effectively the same approach our human brains make by pattern matching. The model is able to group a bunch of features but still has no idea what they relate to.

The development of a speech model may use all of these methods at different stages in the model's training lifecycle. Because there is not one universal speech model, each model should be in a state of continuous training to enhance the accuracy of the model overall or enhance the accuracy for a certain group or use case, if relevant.

Another option in training is to customize the training of a base or enhanced speech model to fit a specific use case or the dialect. For example, if you need a speech model for a McDonald's ordering voicebot, you'll need to address the fact that specific food names are not frequently represented in normal conversation, making them challenging for the model to learn from the data it typically sees.

To solve this problem, you would take McDonald's real-world ordering audio and use that to train a specific McDonald's speech model that understands "Big Mac", "McFlurry", and "Filet-O-Fish". This would make sure the speech model understands the unique food terms and can pick the correct words, which is great for McDonald's but not for Taco Bell or a UK McDonald's, which may have different food items.

Iterating to Create Multiple Models

The McDonald's example raises an important point: no two speech models are the same. Each speech model has different attributes that may work better for some use cases but poorly for others. In the past, most ASR providers only had one speech model in each language because of the difficulty of finding or creating enough of the right data and then training the different interlocking models. Legacy speech models still face challenges with data creation and, to some extent, model training. However, E2EDL models are much more flexible. E2EDL, using transfer link and a data-centric AI approach, can quickly create high accuracy models for new languages, multi-language conversations, or specific use cases within days or weeks.

So how do you quickly build new speech models? For all new speech models, you will need to have the right real-world audio data to use and the ability to use transfer learning. As mentioned above, transfer learning is a way of taking a model that works well for one language or situation, and using it as a seed for a new model so you don't have to start from scratch each time. This effectively transplants all the knowledge from a previous speech model DNN to a new speech model DNN. This method also works in developing new languages, so you are not starting from scratch but have prebuilt knowledge in the DNN that can be used to "learn" a new language.

Next, you take the original model and feed it real-world audio data. This might be food ordering audio in Canadian French, consumer banking audio in Dutch, or news broadcast audio in Ukrainian—whatever it is you want a model to provide transcripts for. This new data is then used by the DNN to create a new speech model that is more accurate in that language and use case than any all-in-one language model.

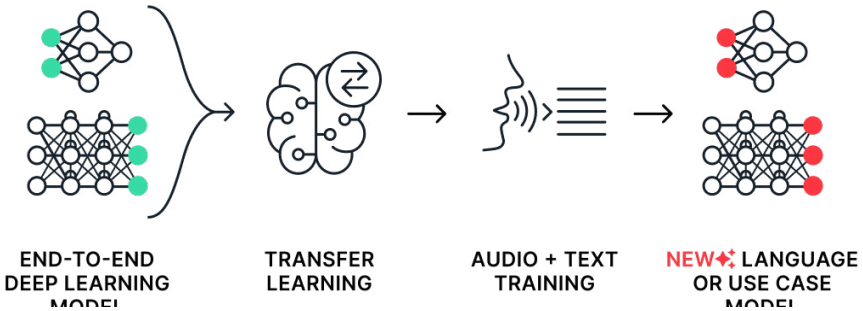


Figure 4. How Deepgram uses transfer learning and curated data to create new speech models.

3 The Impact of End-to-End Deep Learning Models

With our understanding of the different types of models and how they're created out of the way, let's turn to looking at a few different ways that models are deployed in the real world to help underscore why having an E2EDL solution is the most beneficial.

Challenges with All-in-One Speech Models

Currently, most ASR providers have general language models and may have a few use case models like medical or command-and-response use cases, and many companies are working on training a "global" or "universal" language speech model. These all-in-one general models are limited in their ability to handle diverse speech patterns, slang, jargon, and other idiosyncrasies of speech while also staying computationally and cost-efficient.

A global English model must be created in such a way that it does not bias the model toward U.S. dialects. Unfortunately, most public and private training data available are in U.S. English, so your initial model will inevitably have a bias towards U.S. English. Then, as you add data and train for other English dialects, you need to keep the accuracy up on U.S. English while also improving it for other dialects. So, you either build larger DNNs, add more computing power and word/language libraries for predictions, or compromise on certain accents or dialects to actually have an acceptable accuracy across global English. Can this be done? It can certainly be tried, but a far simpler, more scalable solution is already available: create a series of specialized models using end-to-end deep learning and transfer learning.

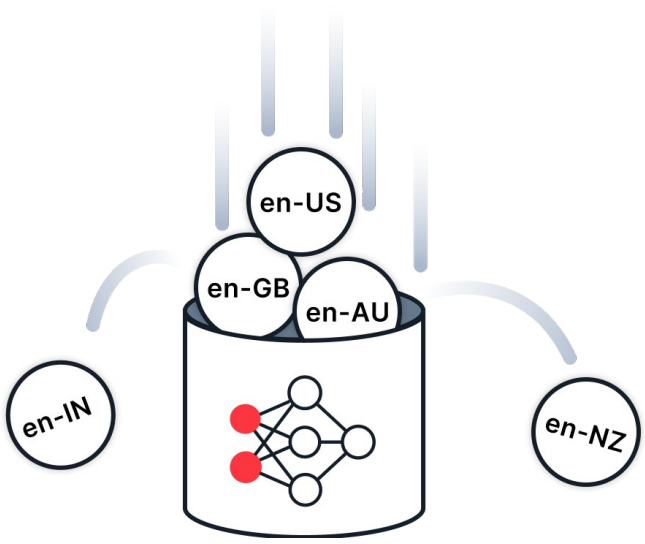


Figure 5. Too much diversity in training inputs lowers accuracy across the board. Model specialization produces the highest accuracy.

The main reason most ASR companies have a global language model is due to their design or their resources. A legacy or poorly designed E2EDL speech model is not flexible enough to quickly and easily create dialect-specific speech models; it could take months or even years. They may not be able to use DNN transfer learning techniques to create new models quickly with the same accuracy as the previous base model.

In addition, many ASR companies are not data-centric AI companies: they focus on the software rather than things like data curation. They don't have in-house linguists, nor data creation or curation resources, so they can only buy expensive language datasets, which will increase their costs and often provide low-quality data.

Another problem with all-in-one models is that they often can't handle the sheer amount of data they would need in certain circumstances. Suppose you want to create a food ordering voicebot for South Africa, where there are 11 official languages and there is a lot of code-switching within a conversation, i.e., switching from English to Afrikaans or Afrikaans to Zulu. Would you want to have a universal speech model that is able to understand all those languages, dialects, accents, and various food names?

From a user perspective, the answer may be "yes"—at least insofar as the user wants one system that can smoothly code-switch between languages the way humans can. But in practical terms, you wouldn't want that solution to be driven by a single universal language model. Creating such a model would require months to years of data collection, training, and possibly a new design. In the end, the speech model will need to cut some corners, especially if it is a legacy-style model. And if the model weighs too heavily on one dialect, language, or accent, then it will have poor accuracy in another dialect, language, or accent.

So what's the solution? **Multiple speech models that directly address the differences in languages, dialects, and use cases**, with a system to dynamically switch between models as it assesses new speech inputs for the highest accuracy.

Multiple Speech Models

The path to better accuracy for all dialects and use cases is to create multiple specialized models. To quickly create new speech models requires E2EDL designed in such a way that it allows flexibility in architecture and transfer learning. In addition, you will need real-world audio and transcribed text data for training in that specific language, dialect, accent, or use case. This work will create a much more accurate speech model for your use case or dialect, as you will not be compromising on one dialect or another.

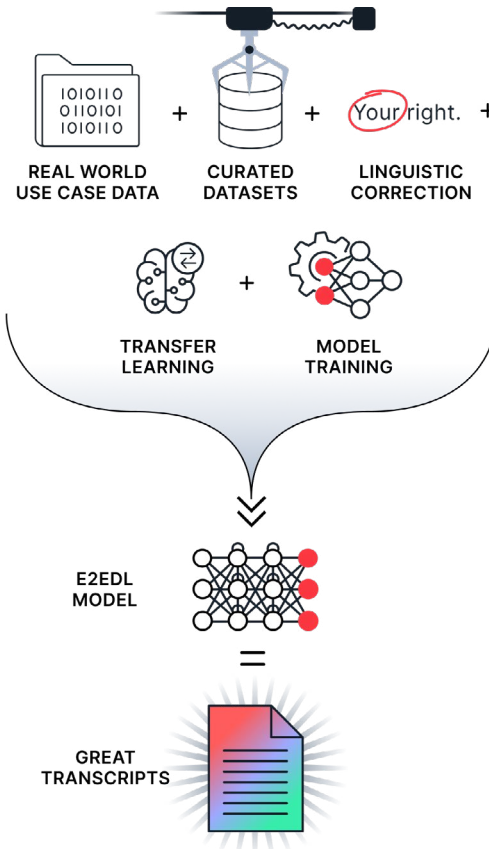


Figure 6. Data-centric AI leverages curated data sets to achieve the highest accuracies.

When do you need multiple speech models?

- The global all-in-one model's accuracy is poor for your use case.
- There are too many different accents or dialects to transcribe correctly with a single model.
- Industry terms, jargon, or acronyms cannot be transcribed correctly.
- The global all-in-one model does not cover a specific dialect that you need.

If you are using multiple speech models, you can use them in different ways. Let's go back to the food ordering example in South Africa. With multiple speech models for your voicebot, you can train one speech model for Afrikaans and Afrikaans accented English for a certain region and train a different speech model for Zulu language regions. In regions where many languages and dialects are spoken, you may even choose a system that dynamically selects the best fit model out of several options.

4 Deepgram Speech Models

Using our data-centric AI approach, Deepgram has built various targeted speech models that better fit different audio environments and use cases. As noted above, this approach creates the most accurate ASR solution for the wide range of audio in the real world. One size never fits all and either causes poor accuracy or slow speed in your speech model.

Base and Enhanced Model Architecture

Deepgram has two foundational speech model architectures that provide a choice between the needs of a particular use case. Our base architecture is built for use with audio in common conversations and has been trained on the most common words used in each language and dialect. Our enhanced architecture is built for audio applications that involve more specific terminology or jargon such as technical meetings, educational lectures, and financial transactions. The enhanced architecture can predict words it has never heard before, so is more accurate for long tail words.

Use Case Models

Deepgram has also trained specific models for various use cases that have different characteristics that can affect audio transcriptions, i.e., environmental noise (phone call or meeting), cross-talk (meeting), and specific terminology (earnings calls). These use-case models are built on top of the two architectures noted above, with more use case models being developed using our curated audio datasets. Our current use case models are available here.

Tailored Models

For more difficult or unique audio characteristics—i.e., thousands of acronyms or terms, accented English from various countries or regions, or static and noise—Deepgram can use a customer's real-world audio data and label it to generate training data. This allows us to specifically match the speech model to the customer's audio characteristics and get the best accuracy possible. Because we are a single end-to-end DNN speech model and have our own data labeling and data generation team, this customization and training can be done within a few weeks and with a little as 10 hours of audio. Of course, the more real-world audio, the better the performance.

5 The Future of Speech Models

The ultimate goal for speech technology is to create a system that can understand all 6,000 languages spoken in the world today. In addition, this system must understand all jargon, slang, industry and academic terminology, acronyms, and accents around the world. To reach this goal and have a useful model, we would need a lot of computing power or every transcript would take a very long time to transcribe.

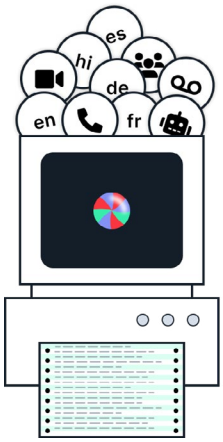


Figure 7. The ultimate multi-language and multi-use case speech recognition system requires too much computing power and has very slow response times.

The other future state could be what we discussed above: multiple speech recognition models tied together by an AI concierge that will choose the right language, use case, or dialect speech model to use depending on the accuracy and other audio features. This AI concierge method could also have a voicebot interface so that you can have a real-life "Jarvis" from the Ironman movies—an AI that understands you and anyone else perfectly, regardless of language, dialect, accent, etc. We have a discussion of this option with our partner OneReach.ai on a webinar panel titled *Improving Human-to-Machine Conversations* that talks about some of these goals.

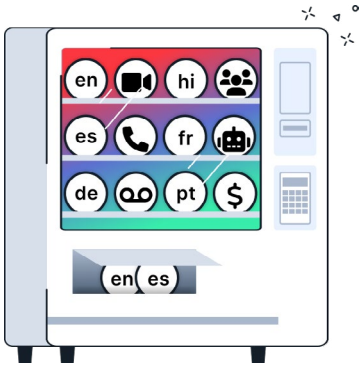


Figure 8. A universal speech recognition solution may be a combination of various AI speech models that are automatically chosen depending on the audio input.

In the foreseeable future, there won't be a universal speech recognition model nor a highly accurate global all-in-one speech model, even for one language. To reach human-level accuracy in speech recognition, the near future requires flexible E2EDL architectures and a data-centric AI focus with transfer learning, real-world audio data, and training to create models that work for the different use cases, dialects, terminology, and jargon that businesses encounter every day. And that's why it matters what's behind the scenes of your speech model—can it keep up? Now? Next year? In two years? Or will it be useful to you as you expand into new languages and regions? Having a modern, end-to-end deep learning speech model that's constantly being improved is your best bet for future-proofing your speech recognition systems.

To learn more about Deepgram's language and speech models, check out our Model Overview, Languages, or Use Cases pages.

About Deepgram

Deepgram is a developer-focused, AI business that is powering the enterprise voice market. Through the Deepgram platform, SDKs, and API, data scientists and developers can convert messy, unstructured audio data into accurate and structured transcriptions in batch or real-time—both on premises and in the cloud. Deepgram builds tailored speech models to optimize their transcription accuracy. Leveraging emerging data, hardware, and software technologies, Deepgram has reduced the speech problem into a single, end-to-end deep learning network enabling customers to reach 95%+ accuracy. In addition to significantly improved accuracy, Deepgram's technology enables better pricing, higher reliability and superior performance at scale.

To learn more visit deepgram.com, get a free API key, or contact us to get started.

DEEPGRAM