

## BLOG ›

# The Machine Learning Behind Hum to Search

THURSDAY, NOVEMBER 12, 2020

*Posted by Christian Frank, Google Research, Zürich*

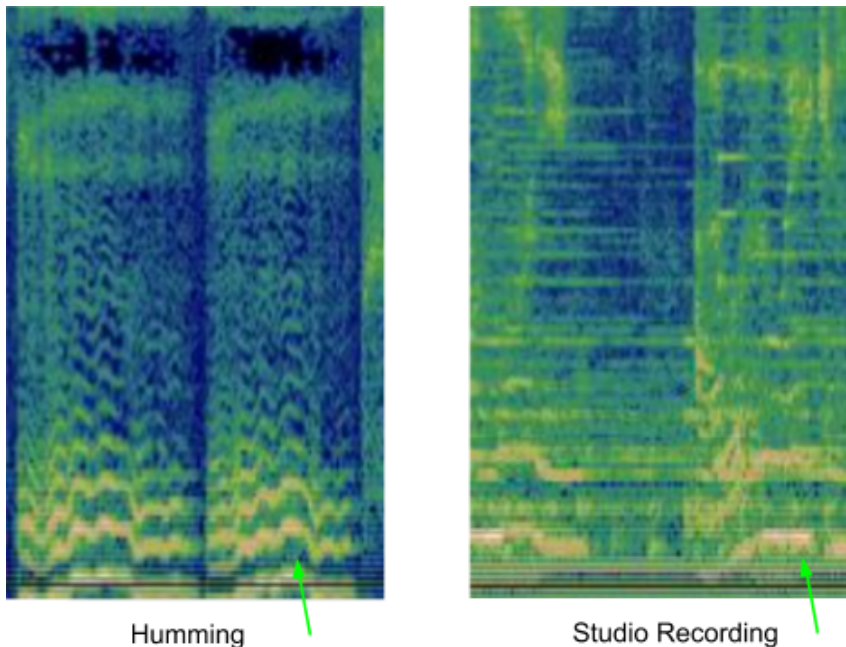
Melodies stuck in your head, often referred to as “[earworms](#),” are a well-known and sometimes irritating phenomenon — once that earworm is there, it can be tough to get rid of it. Research [has found](#) that [engaging with the original song](#), whether that’s listening to or singing it, will drive the earworm away. But what if you can’t quite recall the name of the song, and can only hum the melody?

Existing methods to match a hummed melody to its original polyphonic studio recording face several challenges. With lyrics, background vocals and instruments, the audio of a musical or studio recording can be quite different from a hummed tune. By mistake or design, when someone hums their interpretation of a song, often the pitch, key, tempo or rhythm may vary slightly or even significantly. That’s why so many [existing approaches to query by humming](#) match the hummed tune against a database of pre-existing melody-only or hummed versions of a song, instead of identifying the song directly. However, this type of approach often relies on a limited database that requires manual updates.

Launched in October, [Hum to Search](#) is a new fully machine-learned system within Google Search that allows a person to find a song using only a hummed rendition of it. In contrast to existing methods, this approach produces an embedding of a melody from a spectrogram of a song without generating an intermediate representation. This enables the model to match a hummed melody directly to the original (polyphonic) recordings without the need for a hummed or MIDI version of each track or for other complex hand-engineered logic to extract the melody. This approach greatly simplifies the database for Hum to Search, allowing it to constantly be refreshed with embeddings of original recordings from across the world — even the latest releases.

## Background

Many existing music recognition systems convert an audio sample into a spectrogram before processing it, in order to find a good match. However, one challenge in recognizing a hummed melody is that a hummed tune often contains relatively little information, as illustrated by [this hummed example](#) of [Bella Ciao](#). The difference between the hummed



Visualization of a [hummed clip](#) and a matching studio recording.

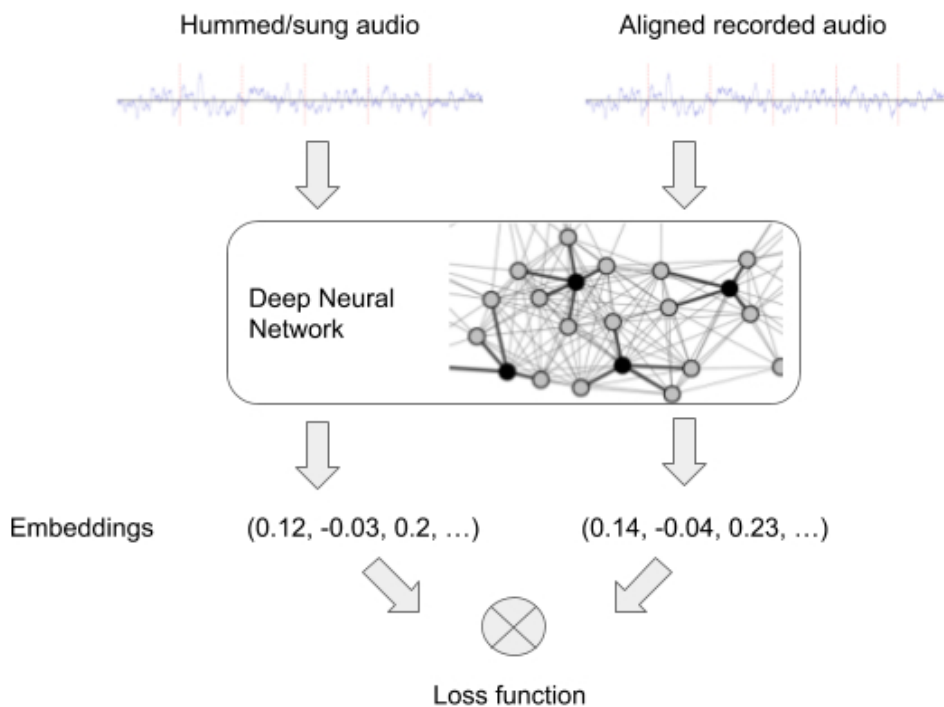
Given the image on the left, a model needs to locate the audio corresponding to the right-hand image from a collection of over 50M similar-looking images (corresponding to segments of studio recordings of other songs). To achieve this, the model has to learn to focus on the dominant melody, and ignore background vocals, instruments, and voice timbre, as well as differences stemming from background noise or room reverberations. To find by eye the dominant melody that might be used to match these two spectrograms, a person might look for similarities in the lines near the bottom of the above images.

Prior efforts to enable discovery of music, in particular in the context of recognizing *recorded* music being played in an environment such as a cafe or a club, demonstrated how machine learning might be applied to this problem. [Now Playing](#), released to Pixel phones in 2017, uses an on-device deep neural network to recognize songs without the need for a server connection, and [Sound Search](#) further developed this technology to provide a server-based recognition service for faster and more accurate searching of over 100 million songs. The next challenge then was to leverage what was learned from these releases to recognize hummed or sung music from a similarly large library of songs.

### Machine Learning Setup

The first step in developing Hum to Search was to modify the music-recognition models used in Now Playing and Sound Search to work with hummed recordings. In principle, many such retrieval systems (e.g., image recognition) work in a similar way. A neural

hummed melody.



*Training setup for the neural network*

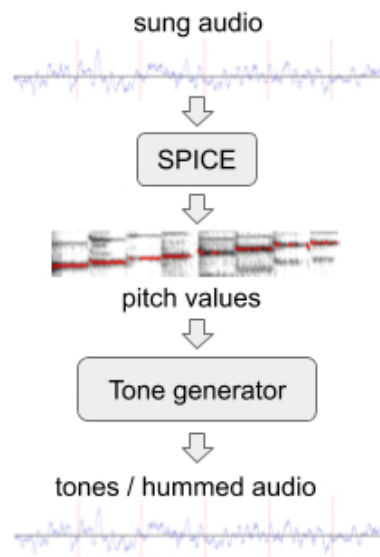
To enable humming recognition, the network should produce embeddings for which pairs of audio containing the same melody are close to each other, even if they have different instrumental accompaniment and singing voices. Pairs of audio containing different melodies should be far apart. In training, the network is provided such pairs of audio until it learns to produce embeddings with this property.

The trained model can then generate an embedding for a tune that is similar to the embedding of the song's reference recording. Finding the correct song is then only a matter of searching for similar embeddings from a database of reference recordings computed from audio of popular music.

### Training Data

Because training of the model required song pairs (recorded and sung), the first challenge was to obtain enough training data. Our initial dataset consisted of mostly sung music segments (very few of these contained humming). To make the model more robust, we augmented the audio during training, for example by varying the pitch or tempo of the sung input randomly. The resulting model worked well enough for people singing, but not for people humming or whistling.

[SPICE](#), a pitch extraction model developed by our wider team as part of the [Neurometric](#) project. SPICE extracts the pitch values from given audio, which we then use to generate a melody consisting of discrete audio tones. The very first version of this system transformed [this original clip](#) into [these tones](#).



*Generating hummed audio from sung audio*

We later refined this approach by replacing the simple tone generator with a neural network that generates audio resembling an actual hummed or whistled tune. For example, the network generates this [humming example](#) or [whistling example](#) from the above [sung clip](#).

As a final step, we compared training data by mixing and matching the audio samples. For example, if we had a similar clip from two different singers, we'd align those two clips with our preliminary models, and are therefore able to show the model an additional pair of audio clips that represent the same melody.

### Machine Learning Improvements

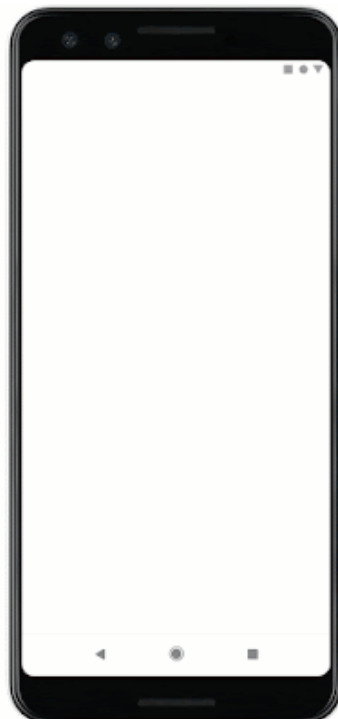
When training the Hum to Search model, we started with a [triplet loss](#) function. This loss [has been shown](#) to perform well across a variety of classification tasks like images and [recorded music](#). Given a pair of audio corresponding to the same melody (points R and P in embedding space, shown below), triplet loss would ignore certain parts of the training data derived from a different melody. This helps the machine improve learning behavior, either when it finds a different melody that is too 'easy' in that it is already far away from R and P (see point E) or because it is too hard in that, given the model's current state of learning, the audio ends up being too close to R — even though according to our data it represents a different melody (see point H).



*Example audio segments visualized as points in embedding space*

We've found that we could improve the accuracy of the model by taking these additional training data (points H and E) into account, namely by formulating a general notion of model confidence across a batch of examples: How sure is the model that *all* the data it has seen can be classified correctly, or has it seen examples that do not fit its current understanding? Based on this notion of confidence, we added a loss that drives model confidence towards 100% across all areas of the embedding space, which led to improvements in our model's [precision and recall](#).

The above changes, but in particular our variations, augmentations and superpositions of the training data, enabled the neural network model deployed in Google Search to recognize sung or hummed melodies. The current system reaches a high level of accuracy on a song database that contains over half a million songs that we are continually updating. This song corpus still has room to grow to include more of the world's many melodies.



say “what's this song?” or click the “Search a song” button, after which you can hum, sing, or whistle away! We hope that Hum to Search can help with that earworm of yours, or maybe just help you in case you want to find and playback a song without having to type its name.

### Acknowledgements

*The work described here was authored by Alex Tudor, Duc Dung Nguyen, Matej Kastelic, Mihajlo Velimirović, Stefan Christoph, Mauricio Zuluaga, Christian Frank, Dominik Roblek, and Matt Sharifi. We would like to deeply thank Krishna Kumar, Satyajeet Salgar and Blaise Aguera y Arcas for their ongoing support, as well as all the Google teams we've collaborated with to build the full Hum to Search product.*

*We would also like to thank all our colleagues at Google who donated clips of themselves singing or humming and therefore laid a foundation for this work, as well as Nick Moukhine for building the Google-internal singing donation app. Finally, special thanks to Meghan Danks and Krishna Kumar for their feedback on earlier versions of this post.*



**Labels:** Audio Machine Learning Search

---

## Previous posts

NOV 12, 2020

Improving On-Device  
Speech Recognition

NOV 9, 2020

OCT 30, 2020

Background Features  
in Google Meet,