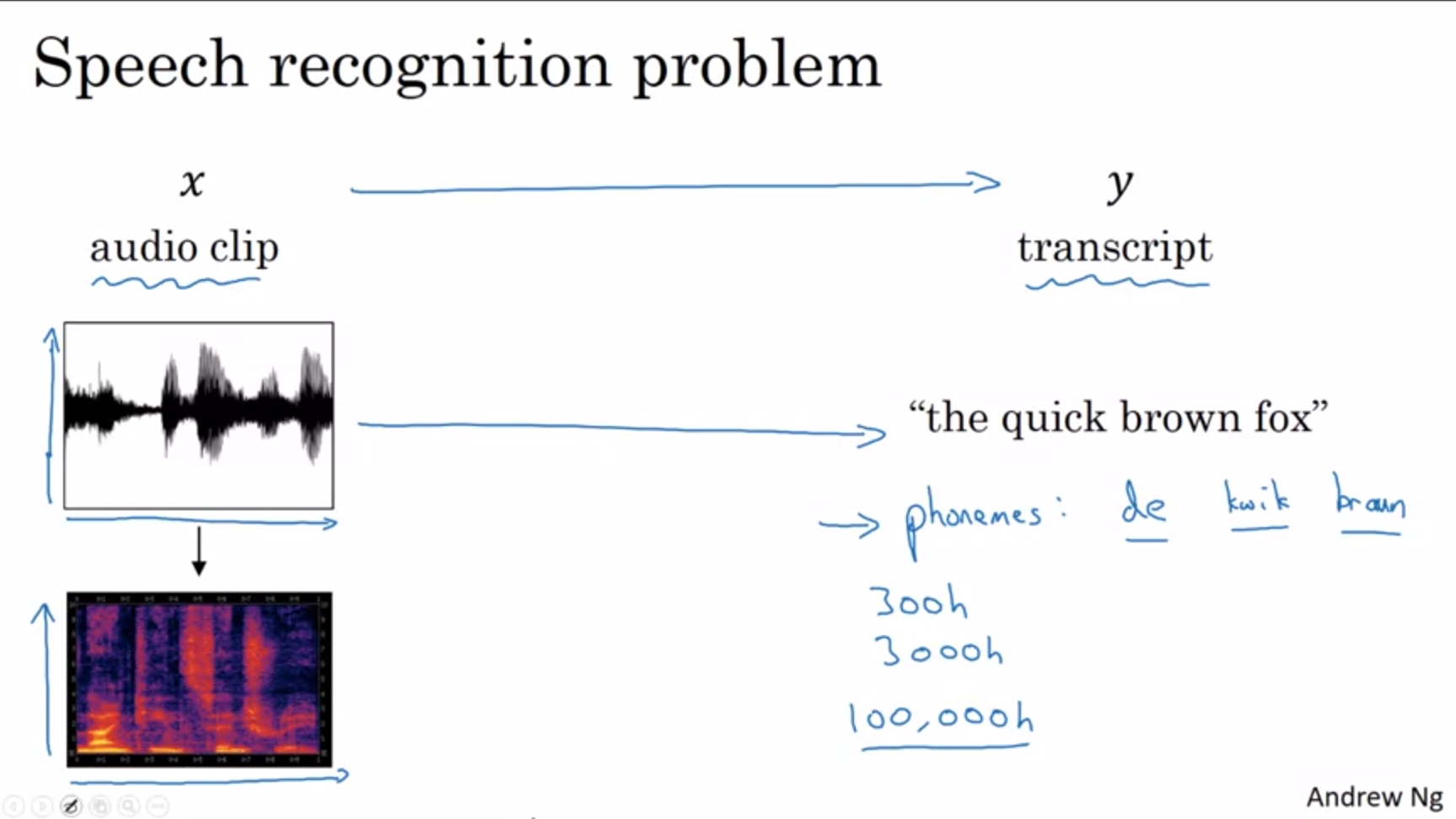


One of the most exciting developments with sequence to sequence models has been the rise of very accurate speech recognition.

We want to have a look at how these sequence to sequence models can be applied to audio data.



The speech recognition problem: given an audio clip x we want to find a transcript y.

A microphone measures minuscular changes in air pressure. The ear is detecting little changes in air pressure.

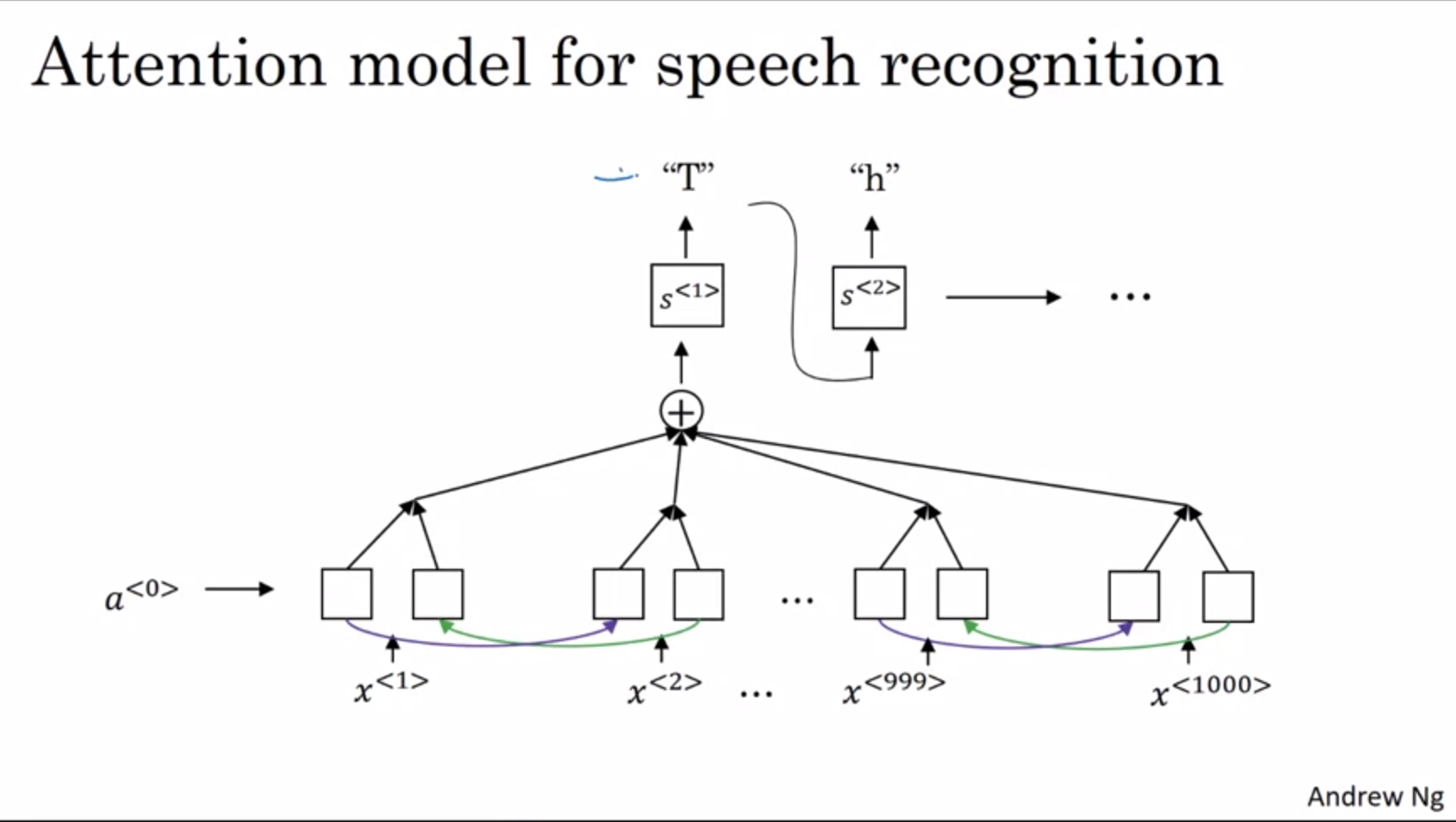
An audio clip plots air pressure against time.

Even the human ear doesn’t process raw wave forms but has physical structures that measures the amount of intensity of different frequencies there are.

A common preprocessing step for audio data is thus to run the raw audio clip and create a spectrogram. A spectrogram is a plot where the horizontal axis is time and the vertical axis is frequencies and intensities of different color show the among of energy (how loud is the sound).

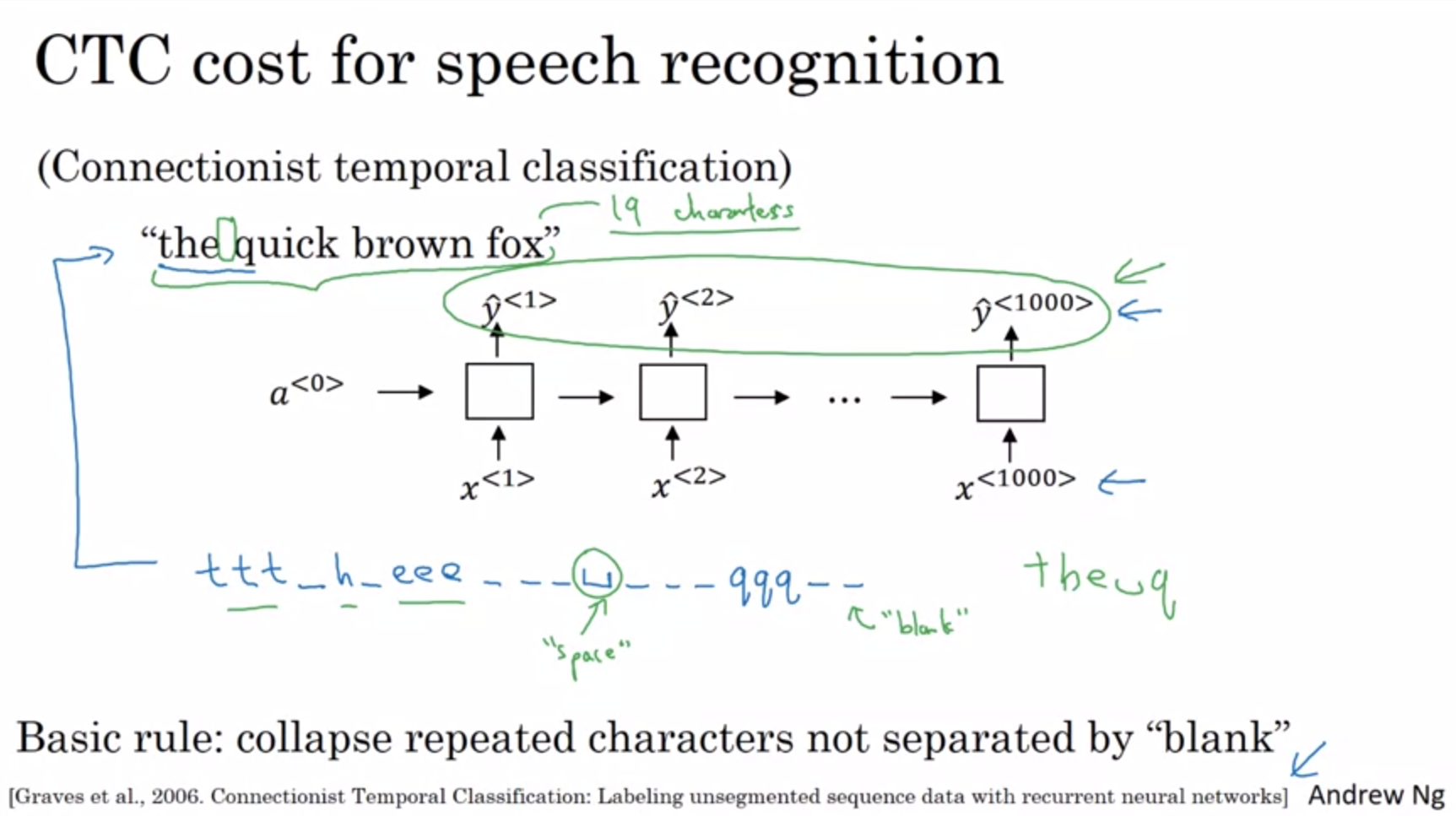
These types of spectrograms are commonly applied preprocessing step. The human ear does a computation pretty similar to this preprocessing step.

One of the exciting trends in speech recognition: once upon a time, phenomes were used. Phenomes: hand-engineered basic units of sound created by linguists. It was thought that these were the best input for speech recognition systems. With end-to-end deep learning we find that phoneme representations are no longer necessary, but we can build speech recognition systems that directly take as input an audio clip and output a transcript. One of the things that made this possible were much larger datasets. Academic datasets on speech recognition might be 300h; 3000h would be considered a reasonable size. The best commercial systems are now trained on 100,000h of audio.



How to build a speech recognition system?

We can build an attention model where on the horizontal axis we take in different timeframes of the audio input and we output the transcript.



One of the methods that works well is the CTC cost for speech recognition. It is due to the paper mentioned on the slide.

Let’s say we have an audio clip “the quick brown fox”. We use an unidirectional RNN with same output size as input size. In practice this would usually be a bidirectional GRU or LSTM and deeper model.

The number of timesteps here is very large. In speech recognition usually the number of input timesteps is much bigger than the number of output timesteps. E.g. 10s of audio with 100Hz -> 1000 inputs. The output might not have 1000 characters.

The CTC cost function allows an output as on the bottom of the slide (this is considered a correct output).

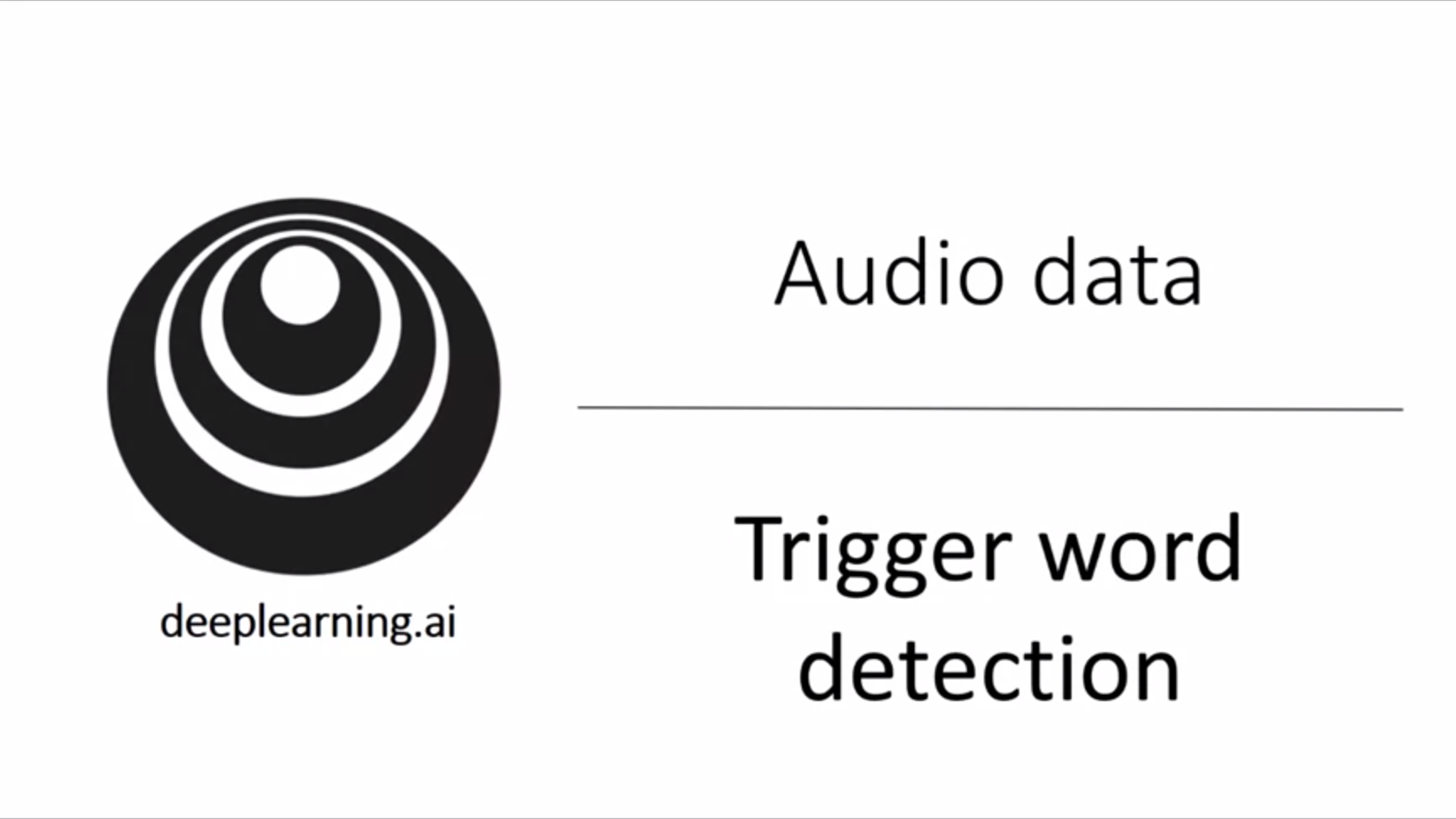
“Blank” is different than the “space” character.

With its basic rule we can also have an output of 1000 characters by repeating the characters and still end up with a shorter output.

Summary: attention models work and CTC models work -> two different options.

Today building an effective production-scale speech recognition system is a pretty significant effort and requires a large dataset.

Now we will look at a trigger word detection system which is much easier to build with a more reasonable amount of data.

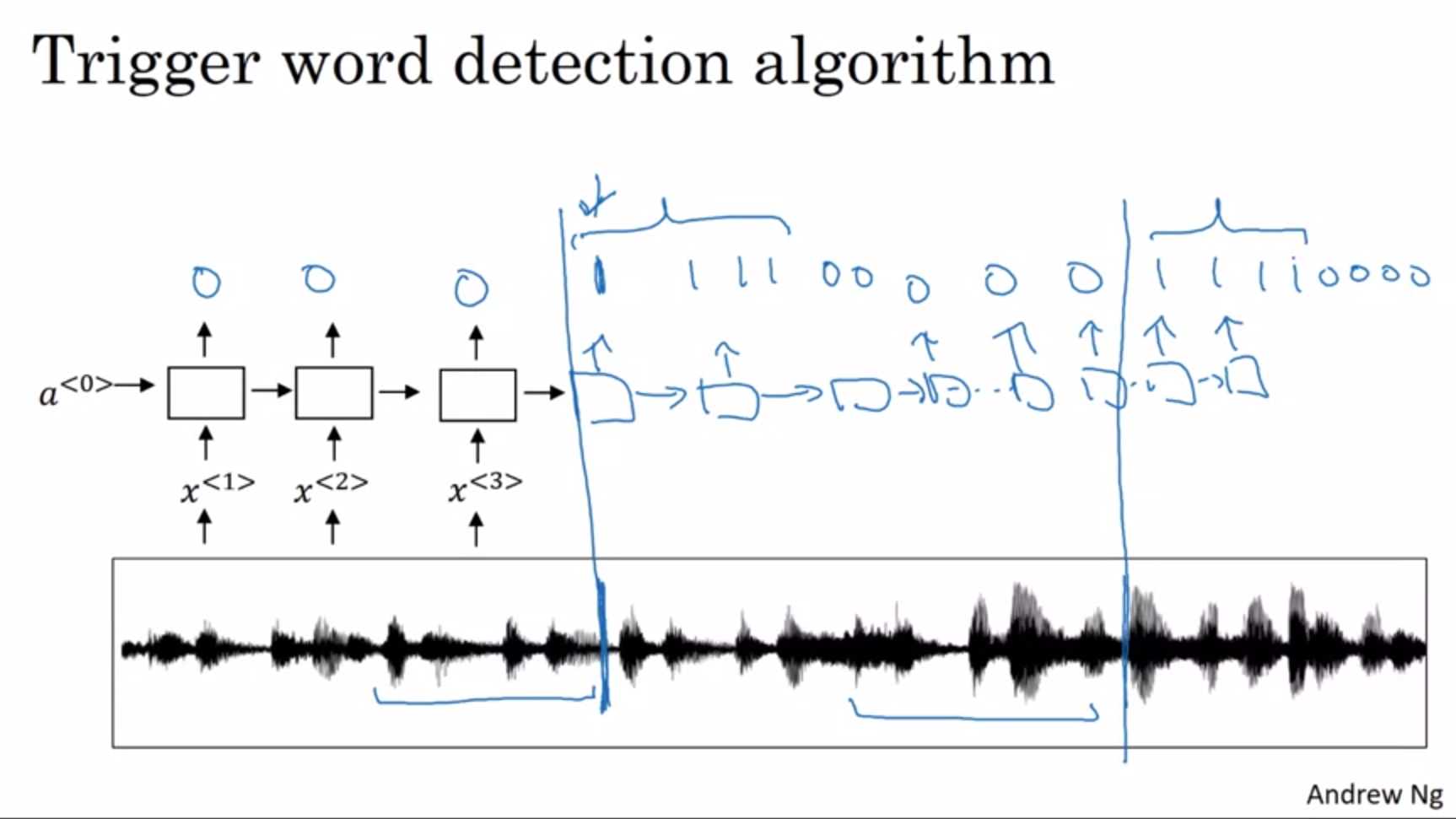


There are more and more devices that we can wake up with our voice -> trigger word detection systems.



These are examples of trigger word detection systems.

If we can build a trigger word detection system we can make our computer do something or we can turn on and off a device.



There is no consensus yet in the literature with regard to what is the best algorithm for trigger word detection systems.

We have raw audio data and use data preprocessing to create a spectrogram. The spectrogram is fed into an RNN.

How to define the target labels y? The marker in the audio clip is when someone said the trigger word. It would work reasonably well if we label the time of the marker as 1. One disadvantage of this is that it creates a very imbalanced dataset. One hack that we can use is instead of setting a single timestep to 1 we can set the output label for a fixed amount of time to 1 before reverting back to 0. This slightly balances out the ratio of 1s and 0s.

Summary: In this course about sequence models we learned about RNNs, GRUs, LSTMs. We learned a lot about word embeddings and how to learn representations of words. We learned about the attention model and how to use it to process audio data.