CS753 - Scientific Article

Speeding up the response time of voice assistants

By Shreya Laddha, Deep Satra and Neilabh Banzal

Speech is the most natural mode of communication for us. And now, computers are getting better and better at communicating with us via speech, to the point that in the next few years, we expect to reach a point where computers will be able to understand speech by anyone - whether its accented or spoken in multiple languages, and will be able to speak with us naturally, to the point that it may not be possible to distinguish between synthesised and natural speech.

The field of computer science that deals with this seemingly simple problem is called "Automatic Speech Recognition" (ASR). However, this is not a trivial task - after all, a baby takes years to learn how to speak and understand a language. We learn to separate the relevant audio from the background noise, to understand people who are speaking at different speeds and in different accents, to understand natural and unnatural pauses and sentence restarts, to make sense of conversations where multiple people are speaking around you, to distinguish between homophones, and so much more. Getting a computer to do all this sounds incredibly complicated, which is why you have to be incredibly smart about ways to recognise the text without all the irrelevant information about the speaker that affects the speak - age, gender, accent, style, etc.

In recent years, heavy strides have been made into making a future with robots speaking naturally and being able to understand us a reality instead of being science fiction.

Most traditional ASR models consist of an Acoustic Feature Generator that converts the audio signal into acoustic features. Then, the decoder uses the Acoustic Model, Pronunciation Model, and Language Model to come up with a guess of what was said in the audio. And this is the output of the ASR system.

1 End-to-end Models

However, in recent years, there has been an end-to-end (E2E) approach towards the speech recognition problem. Here, instead of using multiple algorithms and models, there is a single neural network that encompasses all the models.

The E2E models are essentially "sequence transducers", as they take in a sequence of acoustic features and output a sequence of words. This is implemented as follows - there is an *encoder* which transforms the acoustic inputs into high level representations and a *decoder* which produces linguistic outputs from the encoded representations. One of the major challenges here is that input and output feature lengths differ in lengths, and furthermore, both lengths are variable.

Thus, there are 2 aspects to any transducer - it needs to learn the mapping between the acoustic inputs and the linguistic outputs and at the same time, also learn the alignment between the 2 - which acoustic inputs should correspond to which linguistic outputs?

Usually, alignments between the acoustic inputs and linguistic outputs are unavailable. So neural transducers have to learn both the classification from acoustic features to linguistic predictions as well as the alignment between them.

For this purpose, many neural network-based speech models have been proposed. The 3 most popular ones are - Connectionist Temporal Classification (CTC), RNN-Transducer (RNN-T) and attention-based sequence-to-sequence (Seq2Seq) models. Each of these is built on some foundational assumptions, which are key to their understanding.

CTC makes the assumption that predictions at different time steps are conditionally independent. This is *not* a reasonable assumption for an ASR model, as there is always temporal coherence in speech. This leads to incorrect outputs like "A night died" where the correct output would have been "A knight died".

CTC and RNN-T assume that the alignment between the input and output features is monotonic. For the purpose of the ASR task, which allows us to do streaming transcription, this is a reasonable assumption. However, for a task like audio summarisation, this may not be a reasonable assumption.

Amongst these, RNN-T are expected to perform the best theoretically for the ASR task, based on the assumptions discussed above. They have a further advantage in the sense that the decoding procedure is the simplest amongst the three and the number of hyperparameters to tune is small.

2 RNN-Transducers

RNN Transducers marginalise over all possible alignments. This means that the algorithms involving RNN-Transducers search over the space of all possible alignments and choose the ones that perform the best on the train dataset. Further, they model the dependency between outputs at a particular timesteps as a function of the current state (high level representations) and the previous outputs as opposed to CTC, where the outputs are a function of only the current state.

Decoding in RNN-T base models is done via a beam search algorithm. In a paper presented in May 2020 at ICASSP, a leading conference, George Saon, Zoltan Tuske and Kartik Audhkhasi of IBM Research AI provided an improved beam search which performs better than the state-of the-art Decoding algorithm. The paper is titled *"Alignment-Length Synchronous Decoding for RNN Transducer"*.

3 ALSD vs TSD

We look at 2 decoding algorithms - the Time Synchronous Decoding (TSD) algorithm and the Alignment-Length Synchronous Decoding (ALSD) algorithm. The former is the current state-of-the-art in the ASR community, and the latter is the improvement suggested by Saon, Tuske and Audhkhasi.

The search space for the decoding algorithm is over all possible combinations of the input and the output timesteps. This means that if the input timesteps are enumerated from 1 through T and the output timesteps are enumerated from 1 through U, then we end up with an $T \times U$ grid. And this grid is our search space. The two algorithms are represented in Figure 1.

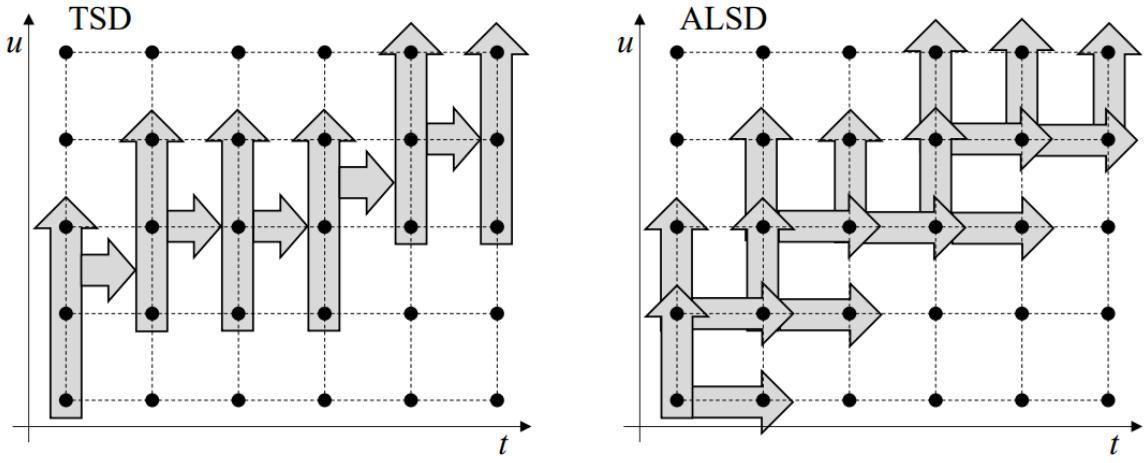

Figure 1: (a) Time Synchronous search space (b) Alignment-Length Synchronous search space

In each Beam Search decoding algorithm, there are multiple hypothesis that the algorithm chooses from. The TSD algorithm chooses among 4 possible hypotheses while the ALSD algorithm chooses between 2 possible hypothesis to predict the output sequence at current time.

Doing a complexity analysis, TSD has a complexity of $T \times k$ whereas ALSD runs in $T + U_{\max}$ steps. Here, $U_{\max}$ is an estimate of maximum output sequence length which is a fraction of T that depends on the type of output symbols (larger for characters, smaller for words) and k is a hyperparameter that's typically 4. In general, ALSD is expected to run faster than TSD for the same accuracy, according to the theory.

The experiments agree with the theory, as ALSD runs faster than TSD consistently for the same accuracy. For the Switchboard 2000 task, this model was able to outperform all other models except those trained on external Language Models.

4 Summary

The progress in the ASR community is extremely rapid, with small improvements like this one appearing pretty frequently. The day is not far when speaking with a Virtual Assistant as naturally as we speak with other people becomes the norm.

References

[1] Eric Battenberg et al. *Exploring Neural Transducers for End-to-End Speech Recognition*. 2017. DOI: 10.48550/ARXIV.1707.07413. URL: https://arxiv.org/abs/1707.07413.

[2] Alex Graves. *Sequence Transduction with Recurrent Neural Networks*. 2012. DOI: 10.48550/ARXIV.1211.3711. URL: https://arxiv.org/abs/1211.3711.

[3] Bjorn Hoffmeister. *Amazon's new research on Automatic Speech recognition*. Dec. 2020. URL: https://www.amazon.science/blog/amazons-new-research-on-automatic-speech-recognition.

[4] Mahaveer Jain. *RNN-T based ASR systems - gatech.edu*. URL: https://www.cc.gatech.edu/classes/AY2021/cs7643_spring/assets/L24_rnnt_asr_tutorial_gt.pdf.

[5] George Saon, Zoltán Tüske, and Kartik Audhkhasi. "Alignment-Length Synchronous Decoding for RNN Transducer". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 7804–7808. DOI: 10.1109/ICASSP40776.2020.9053040.

Authors

Shreya Laddha (180070054), Deep Satra (180040030), Neilabh Banzal (170010014)