# Speech Recognition with Deep RNNs

Project Team Members:
Neeraj(171IT226)
Pruthvi Raju D R (171EC236)
Chiru Charan T S (171EC112)
Niranjan Ramesh Rao (171EC130)

## Abstract

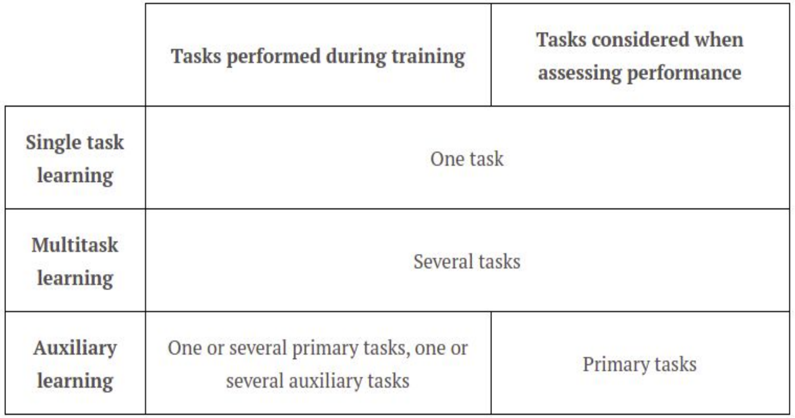
- End-to-end automatic speech recognition (ASR) models with **neural networks** have shown state-of-the-art results compared to **conventional hybrid speech recognizers**.
- **RNN-T** (Recurrent Neural Network Transducer) has shown competitive ASR performance on various benchmarks.
- We apply learning of **auxiliary tasks** to RNN-T based ASR.
- We examine ways in which RNN-T can achieve better accuracy by performing auxiliary tasks.

## Introduction

### Auxiliary Learning:

Auxiliary tasks are tasks we accomplish with the sole objective of better **primary tasks**.

Pseudo reward functions are designed to enable the model to learn the primary task efficiently.



| | **Tasks performed during training** | **Tasks considered when assessing performance** |
|---|---|---|
| **Single task learning** | One task | |
| **Multitask learning** | Several tasks | |
| **Auxiliary learning** | One or several primary tasks, one or several auxiliary tasks | Primary tasks |

## Literature Survey

- [Base paper] Improving RNN Transducer Based ASR with Auxiliary Tasks, IEEE Spoken Language Technology Workshop, 2021, Chunxi Liu et al.
    - Examined the ways to improve accuracy of RNN-T by performing auxiliary tasks
- Deeply Supervised Net, in  Artificial Intelligence and Statistics, 2015 Chen-Yu Lee et al.
    - Improved image recognition with multiple auxiliary classifiers with square hinge losses.
- Self Teaching Networks, Proc. Interspeech, 2019. Liang Lu et al.
    - Improved hybrid ASR models trained with CE criterion.
    - Connected an intermediate layer directly with linear projection layer to compute logits over senomes
- Deja-vu: Double feature presentation and iterated loss in deep transformer networks, Proc. ICASSP, 2020, Andos Tjandra et al.
    - Fed the input features at multiple depths in acoustic model.
    - Improved performance by using the iterated loss.

## Methodology

- Recurrent neural networks (RNNs) are a promising architecture for general-purpose sequence transduction.
- However RNNs are usually restricted to problems where the alignment between the input and output sequence is known in advance.

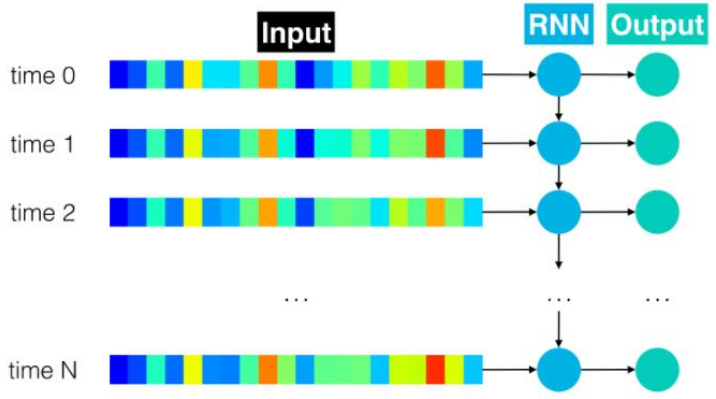
  The problems we're interested in here are **sequence transduction** problems, where the goal is to map an input sequence $x = x_1, x_2, x_3 \ldots\ldots x_n$ to an output sequence $y = y_1, y_2, y_3 \ldots\ldots y_n$

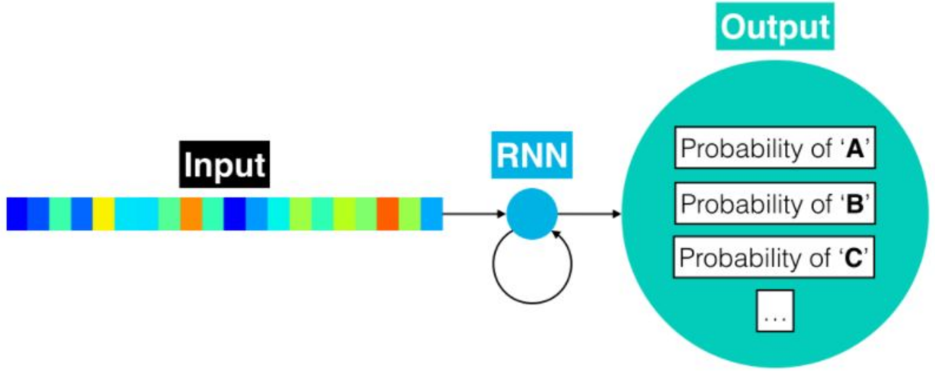
- We will be analysing the performance of 3 types of models :
  - **RNN + TimeDistributed Dense**
  - **Attention-based sequence-to-sequence models**
  - **RNN Transducer**

# Methodology - RNN

At each time step, the speaker pronounces one of 28 possible characters

The output of the RNN at each time step is a vector of probabilities with 29 entries, where the i-th entry encodes the probability that the i-th character is spoken in the time sequence.





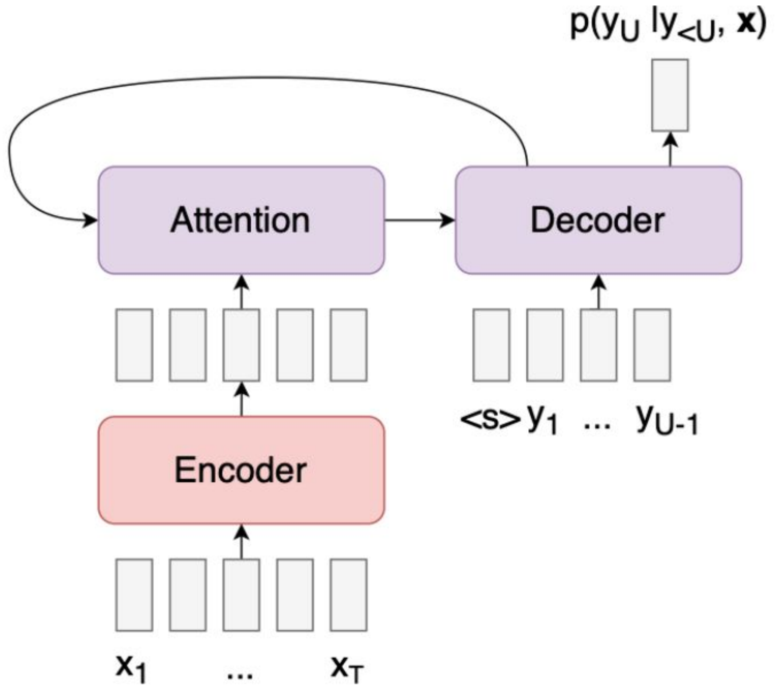
# Attention Models

The model encodes the input *x* into a sequence of feature vectors, then computes the probability of the next output as a function of the encoded input and previous outputs.

The attention mechanism allows the decoder to look at different parts of the input sequence when predicting each output.

**Drawbacks**

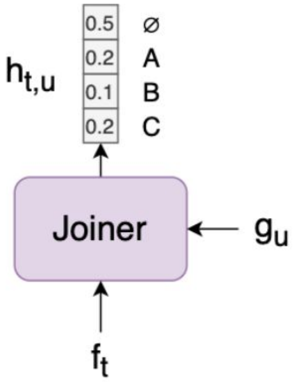
- The attention operation is expensive for long input sequences.
- Attention models don't take advantage of the fact that, for speech recognition, the alignment between inputs and outputs is **monotonic**: if word A comes after word B in the transcript, word A must come after word B in the audio signal

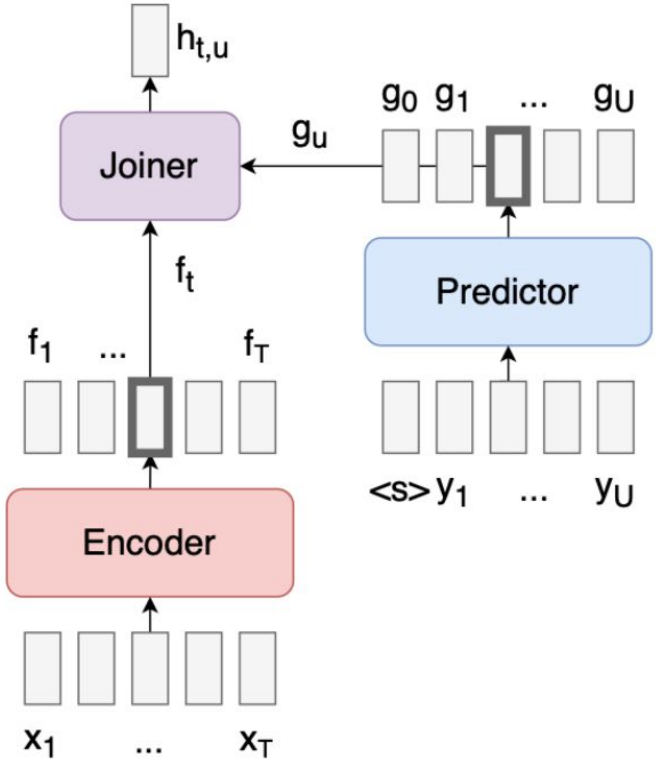


## RNN Transducer

The predictor takes as input the previous outputs and produces features that can be used for predicting the next output, like a standard language model.

The joiner is a simple feedforward network that combines the encoder vector f and predictor vector g and outputs a softmax h over all the labels.





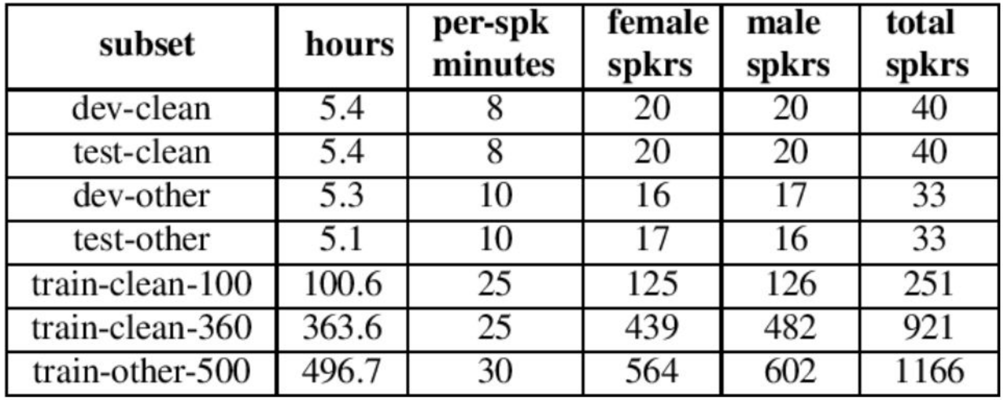
## Dataset And Tool

We will be using the **LibriSpeech** corpus which is a collection of approximately 1,000 hours of audiobooks that are a part of the LibriVox project.

The training data is split into 3 partitions of 100hr, 360hr, and 500hr sets  while the dev and test data are split into the ’clean’ and ’other’ categories.

Each of the dev and test sets is around 5hr in audio length.

Tools/Frameworks used:
Tensorflow, Pytorch

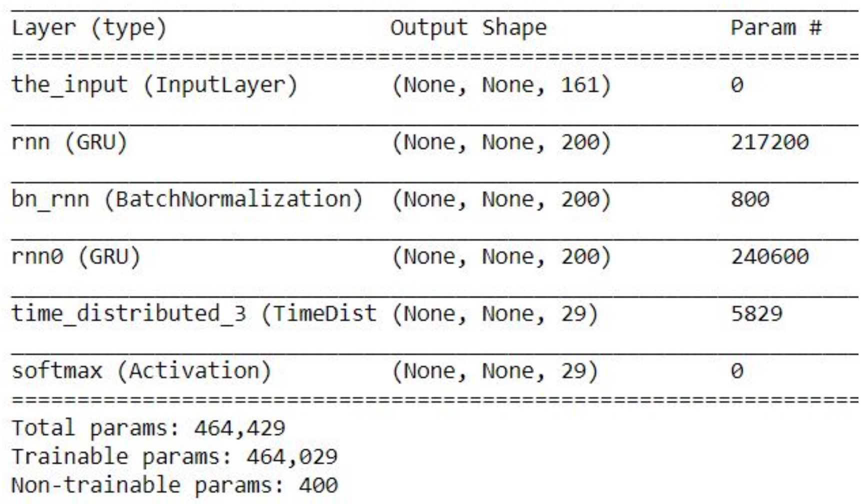


| subset | hours | per-spk minutes | female spkrs | male spkrs | total spkrs |
|---|---|---|---|---|---|
| dev-clean | 5.4 | 8 | 20 | 20 | 40 |
| test-clean | 5.4 | 8 | 20 | 20 | 40 |
| dev-other | 5.3 | 10 | 16 | 17 | 33 |
| test-other | 5.1 | 10 | 17 | 16 | 33 |
| train-clean-100 | 100.6 | 25 | 125 | 126 | 251 |
| train-clean-360 | 363.6 | 25 | 439 | 482 | 921 |
| train-other-500 | 496.7 | 30 | 564 | 602 | 1166 |

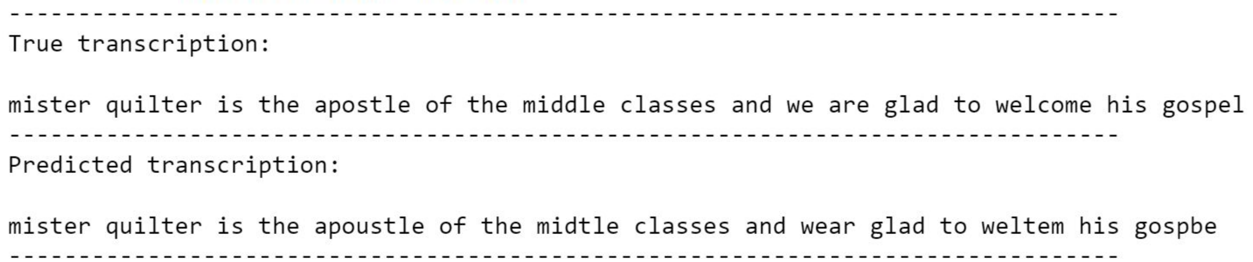
## Progress

Implemented Time Distributed RNN Model with 2 recurrent layers trained with CTC loss function

Extracted **Mel-Frequency Cepstral Coefficients (MFCCs)** as features. MFCC feature is much lower-dimensional than the spectrogram feature, which could help an acoustic model to avoid overfitting to the training dataset.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
the_input (InputLayer)       (None, None, 161)         0
_________________________________________________________________
rnn (GRU)                    (None, None, 200)         217200
_________________________________________________________________
bn_rnn (BatchNormalization)  (None, None, 200)         800
_________________________________________________________________
rnn0 (GRU)                   (None, None, 200)         240600
_________________________________________________________________
time_distributed_3 (TimeDist (None, None, 29)          5829
_________________________________________________________________
softmax (Activation)         (None, None, 29)          0
=================================================================
Total params: 464,429
Trainable params: 464,029
Non-trainable params: 400
```



```
--------------------------------------------------------------------------------
True transcription:

mister quilter is the apostle of the middle classes and we are glad to welcome his gospel
--------------------------------------------------------------------------------
Predicted transcription:

mister quilter is the apoustle of the midtle classes and wear glad to weltem his gospbe
--------------------------------------------------------------------------------
```

## Plan of Work

- [DONE] Implement the Time Distributed RNN Model.
- Implement Attention based sequence to sequence model.
  - Attention is a mechanism combined in the RNN allowing it to focus on certain parts of the input sequence when predicting a certain part of the output sequence, enabling easier learning and of higher quality
- Implement RNN Transducer for the Automatic Speech Recognition task
- Define the auxiliary tasks for auxiliary learning to improve the performance of RNN-T model.
- Train the model on auxiliary model to improve on the primary task (here ASR)
- Analyze the results (Compare the accuracy. Did it improve?)

Thank you