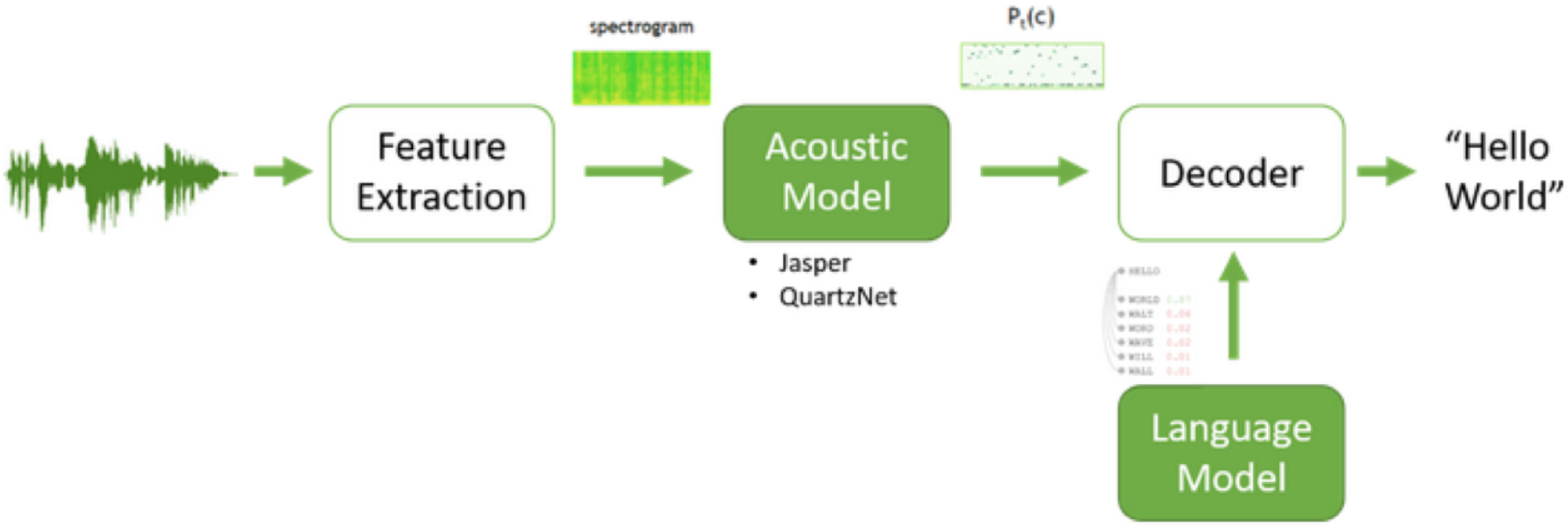
In [1]:
```
from IPython.core.display import display, HTML
display(HTML("<style> *{margin:0; padding:0;} html, body, .container{margin:0;!important padding:0;!importan
t} .container { width:100% !important;}</style>"))
```

End-2-end speech recognititon. Inference. Decoding.

In [2]:
```
import copy
import librosa
import librosa.display
import numpy as np
import math
import os
import matplotlib.pyplot as plt

import nemo
import nemo.collections.asr as nemo_asr
from nemo.collections.asr.helpers import post_process_predictions, post_process_transcripts, word_error_rate
```

```
################################################################################
### WARNING, path does not exist: KALDI_ROOT=/mnt/matylda5/iveselyk/Tools/kaldi-trunk
###          (please add 'export KALDI_ROOT=<your_path>' in your $HOME/.profile)
###          (or run as: KALDI_ROOT=<your_path> python <your_script>.py)
################################################################################
```


spectrogram
$P_t(c)$
Feature Extraction
Acoustic Model
Jasper
QuartzNet
Decoder
"Hello World"
Language Model

In [3]:
```
#just an example of feature extraction
#is not required on further
def preemphasis(signal, coeff=0.97):
    return np.append(signal[0], signal[1:] - coeff * signal[:-1])

def normalize_signal(signal, gain=None):
    if gain is None:
        gain = 1.0 / (np.max(np.abs(signal)) + 1e-5)
    # return signal * gain
    return signal / 2**15-1

def get_speech_features(signal, sample_freq):
    # backend = params.get('backend', 'psf')

    features_type = 'logfbank'
    num_features = 64
    window_size = 20e-3
    window_stride = 10e-3
    augmentation = None

    # if backend == 'librosa':
    window_fn = "hanning"
    dither = 1e-5
    num_fft = 400
    norm_per_feature = True
    mel_basis = None
    gain = None # params.get('gain')
    mean = None # params.get('features_mean')
    std_dev = None # params.get('features_std_dev')
    features, duration = get_speech_features_librosa(
        signal,
        sample_freq,
        num_features,
        features_type,
        window_size,
        window_stride,
        window_fn=window_fn,
        dither=dither,
        num_fft=num_fft,
        norm_per_feature=norm_per_feature,
        mel_basis=mel_basis,
```

```
        gain=gain,
        mean=mean,
        std_dev=std_dev)
    return features, duration

def get_speech_features_librosa(signal,
                                sample_freq,
                                num_features,
                                features_type='spectrogram',
                                window_size=20e-3,
                                window_stride=10e-3,
                                window_fn=np.hanning,
                                num_fft=None,
                                dither=1e-5,
                                norm_per_feature=True,
                                mel_basis=None,
                                gain=None,
                                mean=None,
                                std_dev=None):

    signal = normalize_signal(signal.astype(np.float32), gain)
    signal = signal.astype(np.float32)
    audio_duration = len(signal) * 1.0 / sample_freq

    n_window_size = int(sample_freq * window_size)
    n_window_stride = int(sample_freq * window_stride)
    num_fft = num_fft or 2**math.ceil(math.log2(window_size*sample_freq))

    if dither > 0:
        signal += dither*np.random.randn(*signal.shape)
    # features_type == 'logfbank':
    signal = preemphasis(signal,coeff=0.97)
    S = np.abs(librosa.core.stft(signal, n_fft=num_fft,
                                 hop_length=int(window_stride * sample_freq),
                                 win_length=int(window_size * sample_freq),
                                 center=False, window=window_fn))**2.0

    if mel_basis is None:
      # Build a Mel filter
      mel_basis = librosa.filters.mel(sample_freq,
                                      num_fft,
                                      n_mels=num_features,
```

```
                                        fmin=0,
                                        fmax=int(sample_freq/2))


    features = np.log(np.dot(mel_basis, S) + 1e-20).T
    norm_axis = 0 if norm_per_feature else None
    if mean is None:
        mean = np.mean(features, axis=norm_axis)
    if std_dev is None:
        std_dev = np.std(features, axis=norm_axis)

    features = (features - mean) / std_dev

    return features, audio_duration
```

In [4]:
```
DATA_FOLDER="./wer_1643/wav/"
fn = "br-BR_woman_quiet_close-30cm-_2019-10-15_10-22-20_turn-up-ringer-volume-by-10-.wav"
fn = DATA_FOLDER + fn
sig, sr = librosa.load(fn, sr=16000)
sig *= 32767.
```

In [6]:
```
#wav content
#audio sampled at 16,000 Hz
#16 bits per sample
sig[10000:11000]
```

Out[6]:
```
array([ 195.99402  ,  196.99399  ,  190.99417  ,  196.99399  ,
        201.99384  ,  199.9939   ,  179.9945   ,  190.99417  ,
        195.99402  ,  205.99371  ,  208.99362  ,  200.99387  ,
        196.99399  ,  190.99417  ,  193.99408  ,  192.99411  ,
        202.9938   ,  190.99417  ,  196.99399  ,  197.99396  ,
        195.99402  ,  216.99338  ,  219.99329  ,  225.9931   ,
        224.99313  ,  230.99295  ,  233.99286  ,  232.99289  ,
        238.9927   ,  228.99301  ,  222.9932   ,  217.99335  ,
        207.99365  ,  204.99374  ,  218.99332  ,  211.99353  ,
        209.99359  ,  208.99362  ,  195.99402  ,  195.99402  ,
        182.99442  ,  182.99442  ,  180.99448  ,  184.99435  ,
        173.99469  ,  166.9949   ,  159.99512  ,  152.99533  ,
        156.99521  ,  144.99557  ,  142.99564  ,  136.99582  ,
        137.99579  ,  141.99567  ,  134.99588  ,  124.996185 ,
        128.99606  ,  121.99628  ,  124.996185 ,  121.99628  ,
        118.99637  ,  124.996185 ,  111.99658  ,  103.996826 ,
         94.9971   ,   91.99719  ,   98.99698  ,   90.99722  ,
         83.99744  ,   86.997345 ,   80.99753  ,   85.997375 ,
         83.99744  ,   83.99744  ,   96.99704  ,   83.99744  ,
         88.997284 ,   97.99701  ,   82.99747  ,   88.997284 ,
         86.997345 ,   87.997314 ,   82.99747  ,   93.99713  ,
         82.99747  ,   71.9978   ,   87.997314 ,   74.99771  ,
         93.99713  ,  103.996826 ,  112.99655  ,  107.996704 ,
        114.99649  ,  121.99628  ,  115.99646  ,  128.99606  ,
        121.99628  ,  127.99609  ,  131.99597  ,  148.99545  ,
        152.99533  ,  154.99527  ,  157.99518  ,  157.99518  ,
        171.99475  ,  162.99503  ,  155.99524  ,  161.99506  ,
        164.99496  ,  153.9953   ,  151.99536  ,  139.99573  ,
        137.99579  ,  127.99609  ,  124.996185 ,  140.9957   ,
        131.99597  ,  147.99548  ,  141.99567  ,  130.996    ,
        133.99591  ,  134.99588  ,  142.99564  ,  139.99573  ,
        143.9956   ,  147.99548  ,  143.9956   ,  136.99582  ,
        134.99588  ,  133.99591  ,  125.996155 ,  121.99628  ,
        118.99637  ,  117.9964   ,  115.99646  ,  116.99643  ,
        115.99646  ,  111.99658  ,  108.99667  ,   93.99713  ,
         96.99704  ,   90.99722  ,   82.99747  ,   85.997375 ,
         76.99765  ,   76.99765  ,   66.997955 ,   66.997955 ,
         66.997955 ,   63.998047 ,   62.998077 ,   50.998444 ,
         54.99832  ,   51.998413 ,   47.998535 ,   47.998535 ,
         43.998657 ,   43.998657 ,   37.99884  ,   28.999115 ,
         23.999268 ,   13.999573 ,    2.9999084,    0.9999695,
```

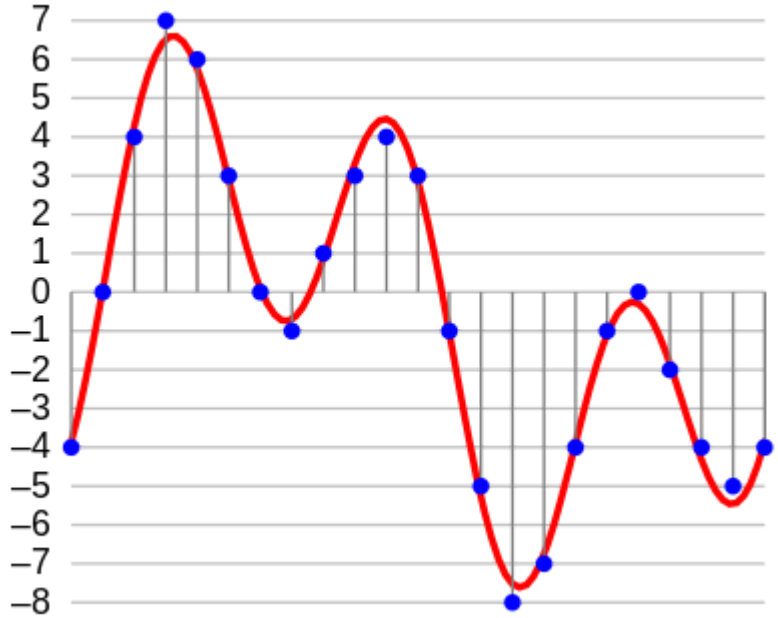
```
  0.       , -11.999634 , -17.99945  , -22.999298 ,
-34.99893  , -32.998993 , -17.99945  , -23.999268 ,
-32.998993 , -29.999084 , -40.99875  , -36.99887  ,
-43.998657 , -48.998505 , -56.99826  , -69.99786  ,
-62.998077 , -75.99768  , -77.99762  , -87.997314 ,
-89.99725  , -85.997375 , -88.997284 , -87.997314 ,
-99.99695  , -98.99698  , -88.997284 , -78.99759  ,
-80.99753  , -79.99756  , -85.997375 , -84.997406 ,
-83.99744  , -83.99744  , -80.99753  , -70.99783  ,
-68.997894 , -72.99777  , -53.998352 , -60.99814  ,
-52.998383 , -44.998627 , -48.998505 , -42.998688 ,
-44.998627 , -34.99893  , -39.99878  , -39.99878  ,
-40.99875  , -40.99875  , -17.99945  , -13.999573 ,
-15.999512 ,   1.999939 ,  10.999664 ,  28.999115 ,
 33.998962 ,  41.99872  ,  58.9982   ,  60.99814  ,
 74.99771  ,  70.99783  ,  67.997925 ,  69.99786  ,
 74.99771  ,  76.99765  ,  68.997894 ,  75.99768  ,
 84.997406 ,  86.997345 ,  99.99695  ,  96.99704  ,
104.996796 , 109.99664  ,  97.99701  , 115.99646  ,
103.996826 ,  99.99695  , 104.996796 ,  92.99716  ,
 86.997345 ,  81.9975   ,  76.99765  ,  65.997986 ,
 65.997986 ,  64.99802  ,  58.9982   ,  58.9982   ,
 64.99802  ,  51.998413 ,  48.998505 ,  50.998444 ,
 49.998474 ,  40.99875  ,  39.99878  ,  50.998444 ,
 27.999146 ,  29.999084 ,  17.99945  ,  18.99942  ,
 21.999329 ,   9.999695 ,  15.999512 ,  13.999573 ,
 16.999481 ,  10.999664 ,   2.9999084,   0.       ,
  8.999725 ,   7.999756 ,  11.999634 ,   8.999725 ,
 17.99945  ,  15.999512 ,   0.       ,   9.999695 ,
  4.9998474,  -1.999939 ,  -8.999725 , -10.999664 ,
-16.999481 , -11.999634 , -14.999542 , -20.99936  ,
-16.999481 , -26.999176 , -26.999176 , -23.999268 ,
-34.99893  , -28.999115 , -33.998962 , -61.998108 ,
-58.9982   , -69.99786  , -79.99756  , -66.997955 ,
-78.99759  , -71.9978   , -67.997925 , -72.99777  ,
-65.997986 , -66.997955 , -58.9982   , -62.998077 ,
-62.998077 , -54.99832  , -54.99832  , -52.998383 ,
-49.998474 , -58.9982   , -52.998383 , -55.99829  ,
-50.998444 , -51.998413 , -54.99832  , -39.99878  ,
-47.998535 , -41.99872  , -47.998535 , -53.998352 ,
-42.998688 , -50.998444 , -59.99817  , -48.998505 ,
-55.99829  , -59.99817  , -67.997925 , -71.9978   ,
```

```
 -71.9978   ,  -68.997894 ,  -60.99814  ,  -62.998077 ,
 -49.998474 ,  -45.998596 ,  -40.99875  ,  -33.998962 ,
 -32.998993 ,  -40.99875  ,  -25.999207 ,  -29.999084 ,
 -36.99887  ,  -33.998962 ,  -45.998596 ,  -37.99884  ,
 -37.99884  ,  -28.999115 ,  -25.999207 ,  -26.999176 ,
 -13.999573 ,   -5.999817 ,   -3.999878 ,   -1.999939 ,
   0.9999695,   -0.9999695,  -14.999542 ,   -6.9997864,
 -13.999573 ,  -27.999146 ,  -21.999329 ,  -25.999207 ,
 -22.999298 ,  -25.999207 ,  -18.99942  ,   -2.9999084,
   6.9997864,    6.9997864,   21.999329 ,   25.999207 ,
  24.999237 ,   26.999176 ,   19.99939  ,   24.999237 ,
  30.999054 ,   33.998962 ,   32.998993 ,   37.99884  ,
  32.998993 ,   40.99875  ,   38.99881  ,   45.998596 ,
  65.997986 ,   63.998047 ,   63.998047 ,   62.998077 ,
  71.9978   ,   68.997894 ,   73.99774  ,   79.99756  ,
  78.99759  ,   70.99783  ,   54.99832  ,   68.997894 ,
  69.99786  ,   52.998383 ,   53.998352 ,   58.9982   ,
  68.997894 ,   83.99744  ,   76.99765  ,   96.99704  ,
 105.996765 ,   95.99707  ,   97.99701  ,   88.997284 ,
  85.997375 ,   79.99756  ,   78.99759  ,   55.99829  ,
  56.99826  ,   38.99881  ,   32.998993 ,   58.9982   ,
  54.99832  ,   64.99802  ,   67.997925 ,   71.9978   ,
  63.998047 ,   63.998047 ,   62.998077 ,   61.998108 ,
  53.998352 ,   42.998688 ,   40.99875  ,   27.999146 ,
  20.99936  ,   10.999664 ,   13.999573 ,    1.999939 ,
   3.999878 ,   12.999603 ,    0.       ,   -5.999817 ,
   2.9999084,    1.999939 ,    2.9999084,   10.999664 ,
  -3.999878 ,  -10.999664 ,  -13.999573 ,  -20.99936  ,
 -34.99893  ,  -40.99875  ,  -61.998108 ,  -60.99814  ,
 -62.998077 ,  -72.99777  ,  -70.99783  ,  -89.99725  ,
 -81.9975   ,  -84.997406 ,  -79.99756  ,  -80.99753  ,
 -85.997375 ,  -91.99719  , -101.99689  , -104.996796 ,
-115.99646  , -120.99631  , -126.996124 , -132.99594  ,
-141.99567  , -131.99597  , -133.99591  , -149.99542  ,
-148.99545  , -141.99567  , -142.99564  , -130.996    ,
-125.996155 , -124.996185 , -116.99643  , -113.99652  ,
-112.99655  , -114.99649  , -108.99667  , -116.99643  ,
-118.99637  , -128.99606  , -126.996124 , -127.99609  ,
-118.99637  , -110.99661  , -108.99667  , -105.996765 ,
-104.996796 ,  -90.99722  , -102.99686  ,  -84.997406 ,
 -84.997406 ,  -92.99716  ,  -88.997284 ,  -98.99698  ,
-100.99692  , -111.99658  , -105.996765 , -101.99689  ,
```

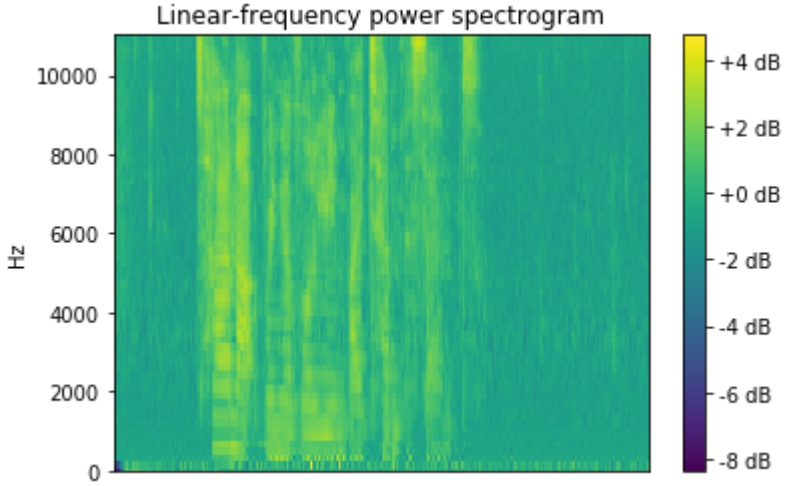
```
-103.996826 ,  -95.99707  ,  -94.9971   ,  -86.997345 ,
 -82.99747  ,  -74.99771  ,  -69.99786  ,  -64.99802  ,
 -56.99826  ,  -54.99832  ,  -56.99826  ,  -65.997986 ,
 -59.99817  ,  -59.99817  ,  -60.99814  ,  -53.998352 ,
 -47.998535 ,  -38.99881  ,  -27.999146 ,  -17.99945  ,
 -12.999603 ,  -17.99945  ,  -15.999512 ,  -16.999481 ,
  -7.999756 ,   -8.999725 ,  -13.999573 ,   -8.999725 ,
 -10.999664 ,    6.9997864,    5.999817 ,   -0.9999695,
  10.999664 ,    7.999756 ,   -1.999939 ,    7.999756 ,
   5.999817 ,    2.9999084,    9.999695 ,    5.999817 ,
  -4.9998474,  -11.999634 ,  -17.99945  ,  -20.99936  ,
 -28.999115 ,  -36.99887  ,  -36.99887  ,  -40.99875  ,
 -43.998657 ,  -44.998627 ,  -35.9989   ,  -36.99887  ,
 -21.999329 ,  -25.999207 ,  -29.999084 ,  -23.999268 ,
 -40.99875  ,  -41.99872  ,  -50.998444 ,  -62.998077 ,
 -62.998077 ,  -66.997955 ,  -79.99756  ,  -65.997986 ,
 -68.997894 ,  -74.99771  ,  -67.997925 ,  -80.99753  ,
 -82.99747  ,  -82.99747  ,  -85.997375 ,  -95.99707  ,
 -97.99701  ,  -99.99695  , -104.996796 , -111.99658  ,
-111.99658  , -120.99631  , -127.99609  , -124.996185 ,
-127.99609  , -126.996124 , -137.99579  , -137.99579  ,
-147.99548  , -151.99536  , -159.99512  , -154.99527  ,
-155.99524  , -165.99493  , -147.99548  , -149.99542  ,
-145.99554  , -147.99548  , -148.99545  , -156.99521  ,
-154.99527  , -155.99524  , -157.99518  , -151.99536  ,
-155.99524  , -152.99533  , -156.99521  , -149.99542  ,
-151.99536  , -153.9953   , -138.99576  , -130.996    ,
-138.99576  , -134.99588  , -133.99591  , -131.99597  ,
-128.99606  , -115.99646  , -104.996796 , -115.99646  ,
-112.99655  , -108.99667  , -107.996704 , -101.99689  ,
 -95.99707  , -108.99667  , -101.99689  ,  -99.99695  ,
-112.99655  , -104.996796 , -107.996704 , -102.99686  ,
-107.996704 , -104.996796 ,  -94.9971   ,  -88.997284 ,
 -92.99716  ,  -80.99753  ,  -80.99753  ,  -90.99722  ,
 -93.99713  , -109.99664  ,  -99.99695  , -105.996765 ,
-105.996765 , -108.99667  , -108.99667  , -102.99686  ,
-104.996796 , -103.996826 , -102.99686  , -102.99686  ,
-102.99686  ,  -96.99704  ,  -95.99707  ,  -92.99716  ,
 -92.99716  ,  -92.99716  ,  -99.99695  ,  -90.99722  ,
-102.99686  , -102.99686  ,  -99.99695  , -113.99652  ,
-105.996765 , -107.996704 , -100.99692  , -107.996704 ,
-112.99655  , -114.99649  , -113.99652  , -107.996704 ,
```

```
-103.996826 , -101.99689  ,  -99.99695  ,  -99.99695  ,
 -92.99716  ,  -93.99713  , -107.996704 , -102.99686  ,
-108.99667  , -124.996185 , -130.996    , -139.99573  ,
-144.99557  , -138.99576  , -139.99573  , -127.99609  ,
-132.99594  , -138.99576  , -138.99576  , -138.99576  ,
-142.99564  , -148.99545  , -145.99554  , -157.99518  ,
-157.99518  , -167.99487  , -175.99463  , -166.9949   ,
-159.99512  , -162.99503  , -153.9953   , -142.99564  ,
-140.9957   , -133.99591  , -135.99585  , -133.99591  ,
-127.99609  , -127.99609  , -140.9957   , -139.99573  ,
-135.99585  , -133.99591  , -132.99594  , -144.99557  ,
-135.99585  , -128.99606  , -131.99597  , -127.99609  ,
-124.996185 , -121.99628  , -118.99637  , -119.99634  ,
-126.996124 , -117.9964   , -116.99643  , -111.99658  ,
 -98.99698  , -104.996796 ,  -87.997314 ,  -81.9975   ,
 -86.997345 ,  -80.99753  ,  -81.9975   ,  -82.99747  ,
 -74.99771  ,  -67.997925 ,  -76.99765  ,  -72.99777  ,
 -65.997986 ,  -67.997925 ,  -54.99832  ,  -46.998566 ,
 -50.998444 ,  -50.998444 ,  -49.998474 ,  -38.99881  ,
 -36.99887  ,  -48.998505 ,  -39.99878  ,  -41.99872  ,
 -37.99884  ,  -39.99878  ,  -41.99872  ,  -30.999054 ,
 -35.9989   ,  -22.999298 ,  -18.99942  ,  -18.99942  ,
  -6.9997864,   -3.999878 ,    5.999817 ,    8.999725 ,
  13.999573 ,   24.999237 ,   22.999298 ,   27.999146 ,
  27.999146 ,   29.999084 ,   34.99893  ,   29.999084 ,
  20.99936  ,   23.999268 ,   22.999298 ,   14.999542 ,
  18.99942  ,   13.999573 ,   12.999603 ,    9.999695 ,
   7.999756 ,   14.999542 ,    9.999695 ,   10.999664 ,
  24.999237 ,   14.999542 ,   13.999573 ,   14.999542 ,
  17.99945  ,   16.999481 ,   26.999176 ,   34.99893  ,
  17.99945  ,   17.99945  ,   15.999512 ,    5.999817 ,
   6.9997864,   12.999603 ,    8.999725 ,    8.999725 ,
   8.999725 ,   12.999603 ,    7.999756 ,    7.999756 ,
  18.99942  ,   10.999664 ,   11.999634 ,   11.999634 ,
   6.9997864,    2.9999084,    1.999939 ,   -1.999939 ,
 -13.999573 ,  -11.999634 ,  -15.999512 ,  -22.999298 ,
 -21.999329 ,  -20.99936  ,  -24.999237 ,  -25.999207 ,
 -25.999207 ,  -22.999298 ,  -24.999237 ,  -27.999146 ,
 -20.99936  ,  -25.999207 ,  -22.999298 ,  -20.99936  ,
 -28.999115 ,  -28.999115 ,  -33.998962 ,  -44.998627 ,
 -46.998566 ,  -51.998413 ,  -47.998535 ,  -46.998566 ,
 -49.998474 ,  -46.998566 ,  -55.99829  ,  -44.998627 ,
```

```
       -41.99872  ,  -44.998627 ,  -38.99881  ,  -33.998962 ,
       -28.999115 ,  -23.999268 ,  -15.999512 ,  -14.999542 ,
        -5.999817 ,    1.999939 ,    0.9999695,    8.999725 ,
        13.999573 ,    1.999939 ,    7.999756 ,    1.999939 ,
        -3.999878 ,   -3.999878 ,   -9.999695 ,   -5.999817 ,
        -3.999878 ,    2.9999084,   15.999512 ,   19.99939  ,
        33.998962 ,   41.99872  ,   39.99878  ,   41.99872  ,
        32.998993 ,   34.99893  ,   29.999084 ,   36.99887  ,
        23.999268 ,   23.999268 ,   16.999481 ,    9.999695 ,
        20.99936  ,    7.999756 ,   25.999207 ,   22.999298 ,
        24.999237 ,   34.99893  ,   34.99893  ,   45.998596 ,
        45.998596 ,   52.998383 ,   49.998474 ,   47.998535 ,
        47.998535 ,   39.99878  ,   37.99884  ,   39.99878  ,
        43.998657 ,   52.998383 ,   45.998596 ,   54.99832  ,
        56.99826  ,   53.998352 ,   59.99817  ,   63.998047 ,
        73.99774  ,   53.998352 ,   63.998047 ,   61.998108 ,
        65.997986 ,   73.99774  ,   57.99823  ,   62.998077 ,
        52.998383 ,   63.998047 ,   58.9982   ,   56.99826  ,
        70.99783  ,   63.998047 ,   74.99771  ,   72.99777  ,
        65.997986 ,   60.99814  ,   55.99829  ,   48.998505 ,
        35.9989   ,   34.99893  ,   25.999207 ,   22.999298 ,
        21.999329 ,   17.99945  ,   19.99939  ,   21.999329 ,
        26.999176 ,   29.999084 ,   28.999115 ,   27.999146 ,
        29.999084 ,   24.999237 ,   21.999329 ,   17.99945  ,
         5.999817 ,    0.9999695,   -5.999817 ,   -9.999695 ,
        -5.999817 ,  -12.999603 ,   -5.999817 ,   -1.999939 ,
         0.       ,    3.999878 ,  -10.999664 ,   -6.9997864,
        -4.9998474,   -9.999695 ,  -10.999664 ,  -13.999573 ,
       -18.99942  ,  -35.9989   ,  -46.998566 ,  -49.998474 ,
       -55.99829  ,  -54.99832  ,  -56.99826  ,  -63.998047 ,
       -64.99802  ,  -66.997955 ,  -73.99774  ,  -57.99823  ,
       -54.99832  ,  -64.99802  ,  -53.998352 ,  -64.99802  ,
       -64.99802  ,  -74.99771  ,  -77.99762  ,  -92.99716  ,
       -95.99707  ,  -94.9971   , -102.99686  ,  -94.9971   ,
      -105.996765 ,  -99.99695  , -100.99692  ,  -87.997314 ,
       -92.99716  ,  -90.99722  ,  -89.99725  ,  -94.9971   ,
       -87.997314 ,  -84.997406 ,  -85.997375 ,  -84.997406 ,
       -86.997345 ,  -88.997284 ,  -89.99725  , -104.996796 ,
       -94.9971   , -110.99661  , -104.996796 ,  -96.99704  ,
       -99.99695  ,  -85.997375 ,  -79.99756  ,  -78.99759  ,
       -79.99756  ,  -60.99814  ,  -52.998383 ,  -54.99832  ],
      dtype=float32)
```


7
6
5
4
3
2
1
0
−1
−2
−3
−4
−5
−6
−7
−8

In [7]:
```
ret, sr = get_speech_features(sig, sr)
librosa.display.specshow(ret.T, cmap='viridis', y_axis='linear')
plt.colorbar(format='%+2.0f dB')
plt.title('Linear-frequency power spectrogram')
#plt.title('Spectrogram')
plt.show()
```



NeMo

https://github.com/NVIDIA (https://github.com/NVIDIA)

In [8]:
```
MODEL_YAML = './wer_1643/quartznet15x5.yaml'
CHECKPOINT_ENCODER = './wer_1643/JasperEncoder-STEP-37750.pt'
CHECKPOINT_DECODER = './wer_1643/JasperDecoderForCTC-STEP-37750.pt'
```

In [9]:
```
neural_factory = nemo.core.NeuralModuleFactory(
    placement=nemo.core.DeviceType.CPU,
    backend=nemo.core.Backend.PyTorch)
```

In [10]:
```
from ruamel.yaml import YAML
#parce config file
yaml = YAML(typ="safe")
with open(MODEL_YAML) as f:
    model_definition = yaml.load(f)

vocab = model_definition['labels']
sample_rate = model_definition['sample_rate']
```

In [11]:
```
model_definition
```

Out[11]:

```
{'model': 'QuartzNet',
 'sample_rate': 16000,
 'AudioToTextDataLayer': {'max_duration': 16.7,
  'trim_silence': True,
  'train': {'shuffle': True},
  'eval': {'shuffle': False, 'max_duration': None}},
 'AudioToMelSpectrogramPreprocessor': {'window_size': 0.02,
  'window_stride': 0.01,
  'window': 'hann',
  'normalize': 'per_feature',
  'n_fft': 512,
  'features': 64,
  'dither': 1e-05,
  'pad_to': 16,
  'stft_conv': True},
 'SpectrogramAugmentation': {'rect_masks': 5,
  'rect_time': 120,
  'rect_freq': 50},
 'JasperEncoder': {'activation': 'relu',
  'conv_mask': True,
  'jasper': [{'filters': 256,
    'repeat': 1,
    'kernel': [33],
    'stride': [2],
    'dilation': [1],
    'dropout': 0.0,
    'residual': False,
    'separable': True},
   {'filters': 256,
    'repeat': 5,
    'kernel': [33],
    'stride': [1],
    'dilation': [1],
    'dropout': 0.0,
    'residual': True,
    'separable': True},
   {'filters': 256,
    'repeat': 5,
    'kernel': [33],
    'stride': [1],
    'dilation': [1],
```

```
 'dropout': 0.0,
 'residual': True,
 'separable': True},
{'filters': 256,
 'repeat': 5,
 'kernel': [33],
 'stride': [1],
 'dilation': [1],
 'dropout': 0.0,
 'residual': True,
 'separable': True},
{'filters': 256,
 'repeat': 5,
 'kernel': [39],
 'stride': [1],
 'dilation': [1],
 'dropout': 0.0,
 'residual': True,
 'separable': True},
{'filters': 256,
 'repeat': 5,
 'kernel': [39],
 'stride': [1],
 'dilation': [1],
 'dropout': 0.0,
 'residual': True,
 'separable': True},
{'filters': 256,
 'repeat': 5,
 'kernel': [39],
 'stride': [1],
 'dilation': [1],
 'dropout': 0.0,
 'residual': True,
 'separable': True},
{'filters': 512,
 'repeat': 5,
 'kernel': [51],
 'stride': [1],
 'dilation': [1],
 'dropout': 0.0,
 'residual': True,
```

```
                    'separable': True},
                   {'filters': 512,
                    'repeat': 5,
                    'kernel': [51],
                    'stride': [1],
                    'dilation': [1],
                    'dropout': 0.0,
                    'residual': True,
                    'separable': True},
                   {'filters': 512,
                    'repeat': 5,
                    'kernel': [51],
                    'stride': [1],
                    'dilation': [1],
                    'dropout': 0.0,
                    'residual': True,
                    'separable': True},
                   {'filters': 512,
                    'repeat': 5,
                    'kernel': [63],
                    'stride': [1],
                    'dilation': [1],
                    'dropout': 0.0,
                    'residual': True,
                    'separable': True},
                   {'filters': 512,
                    'repeat': 5,
                    'kernel': [63],
                    'stride': [1],
                    'dilation': [1],
                    'dropout': 0.0,
                    'residual': True,
                    'separable': True},
                   {'filters': 512,
                    'repeat': 5,
                    'kernel': [63],
                    'stride': [1],
                    'dilation': [1],
                    'dropout': 0.0,
                    'residual': True,
                    'separable': True},
                   {'filters': 512,
```

```
                'repeat': 5,
                'kernel': [75],
                'stride': [1],
                'dilation': [1],
                'dropout': 0.0,
                'residual': True,
                'separable': True},
               {'filters': 512,
                'repeat': 5,
                'kernel': [75],
                'stride': [1],
                'dilation': [1],
                'dropout': 0.0,
                'residual': True,
                'separable': True},
               {'filters': 512,
                'repeat': 5,
                'kernel': [75],
                'stride': [1],
                'dilation': [1],
                'dropout': 0.0,
                'residual': True,
                'separable': True},
               {'filters': 512,
                'repeat': 1,
                'kernel': [87],
                'stride': [1],
                'dilation': [2],
                'dropout': 0.0,
                'residual': False,
                'separable': True},
               {'filters': 1024,
                'repeat': 1,
                'kernel': [1],
                'stride': [1],
                'dilation': [1],
                'dropout': 0.0,
                'residual': False}]},
              'labels': [' ',
               'a',
               'b',
               'c',
```

```
            'd',
            'e',
            'f',
            'g',
            'h',
            'i',
            'j',
            'k',
            'l',
            'm',
            'n',
            'o',
            'p',
            'q',
            'r',
            's',
            't',
            'u',
            'v',
            'w',
            'x',
            'y',
            'z',
            "'"]}
```

In [12]:
```
eval_datasets = '/home/artsokol/wer_1643/recordings.json'
```

In [13]:
```
!cat /home/artsokol/wer_1643/recordings.json
```

```
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_man_quiet_close-30cm-_2019-10-08_16-38-59_play-breakin
g-bad.wav", "text": "play breaking bad", "duration": 4.4}
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_woman_quiet_close-30cm-_2019-10-15_10-22-20_turn-up-ri
nger-volume-by-10-.wav", "text": "turn up ringer volume by 10 persent", "duration": 4.4}
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_man_quiet_close-30cm-_2019-10-08_17-00-39_call-jack-on
-his-uk-number.wav","text":"call jack on his uk number","duration":4.84}
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_woman_quiet_close-30cm-_2019-10-15_13-12-34_turn-off-t
he-monday-alarm.wav","text":"turn off monday alarm","duration":4.88}
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_woman_quiet_middle-80cm-_2019-10-15_14-59-28_call-0143
9454544.wav","text":"call one four three nine five four five four four","duration":6.16}
{"audio_filepath": "/home/artsokol/wer_1643/wav/br-BR_man_quiet_close-30cm-_2019-10-08_17-44-38_call-amelia.
wav","text":"call amelia","duration":2.36}
```

In [14]:
```
eval_dl_params = copy.deepcopy(model_definition["AudioToTextDataLayer"])
eval_dl_params.update(model_definition["AudioToTextDataLayer"]["eval"])
del eval_dl_params["train"]
del eval_dl_params["eval"]
data_layer = nemo_asr.AudioToTextDataLayer(
    manifest_filepath=eval_datasets,
    sample_rate=sample_rate,
    labels=vocab,
    batch_size=64,
    **eval_dl_params,
)
```

```
[NeMo I 2020-05-19 18:28:12 collections:142] Dataset loaded with 6 files totalling 0.01 hours
[NeMo I 2020-05-19 18:28:12 collections:143] 0 files were filtered totalling 0.00 hours
```

In [15]:
```
logging = nemo.logging
```

In [16]:
```
data_preprocessor = nemo_asr.AudioToMelSpectrogramPreprocessor(
        sample_rate=sample_rate, **model_definition["AudioToMelSpectrogramPreprocessor"],
)
```

```
[NeMo I 2020-05-19 18:28:56 features:144] PADDING: 16
[NeMo I 2020-05-19 18:28:56 features:152] STFT using conv
```

In [17]:
```
#create encoder
jasper_encoder = nemo_asr.JasperEncoder(
    feat_in=model_definition["AudioToMelSpectrogramPreprocessor"]["features"], **model_definition["JasperEnco
der"]
)
```

In [18]:
```
#create decoder
jasper_decoder = nemo_asr.JasperDecoderForCTC(
    feat_in=model_definition["JasperEncoder"]["jasper"][-1]["filters"], num_classes=len(vocab)
)
```

In [19]:
```
greedy_decoder = nemo_asr.GreedyCTCDecoder()

# labels = [" ", "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m",
#           "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z", "'", "_"]
# def greedy_decoder(logits, vocab=labels, merge=True):
#     s = ''
#     c = ''
#     for i in range(logits.shape[1]):
#         c_i = vocab[np.argmax(logits[0,i])]
#         if merge and c_i == c:
#             continue
#         s += c_i
#         c = c_i
#     if merge:
#         s = s.replace('_', '')
#         # pass
#     print(s)
```

In [20]:
```
audio_signal, a_sig_length, transcript, transcript_len = data_layer()
processed_signal, p_length = data_preprocessor(input_signal=audio_signal, length=a_sig_length)
```

In [21]:
```
encoded, encoded_len = jasper_encoder(audio_signal=processed_signal, length=p_length)
```

In [22]:
```
log_probs = jasper_decoder(encoder_output=encoded)
```

In [23]:
```
predictions = greedy_decoder(log_probs=log_probs)
```

In [24]:
```
eval_tensors = [log_probs, predictions, transcript, transcript_len, encoded_len]
```

In [25]:
```
evaluated_tensors = neural_factory.infer(tensors=eval_tensors, checkpoint_dir='./wer_1643')
```

```
[NeMo I 2020-05-19 18:31:03 actions:1493] Restoring JasperEncoder from ./wer_1643/JasperEncoder-STEP-37750.p
t
[NeMo I 2020-05-19 18:31:03 actions:1493] Restoring JasperDecoderForCTC from ./wer_1643/JasperDecoderForCTC-
STEP-37750.pt
[NeMo I 2020-05-19 18:31:03 actions:734] Evaluating batch 0 out of 1

/pytorch/aten/src/ATen/native/BinaryOps.cpp:81: UserWarning: Integer division of tensors using div or / is d
eprecated, and in a future release div will perform true division as in Python 3. Use true_divide or floor_d
ivide (// in Python) instead.
```

In [115]:
```
evaluated_tensors
```

Out[115]:
```
[[tensor([[[-1.1825e+01, -1.1150e+01, -1.0174e+01,  ..., -1.5053e+01,
            -1.2467e+01, -2.8094e-04],
           [-1.1902e+01, -1.1296e+01, -1.0591e+01,  ..., -1.5041e+01,
            -1.2617e+01, -2.6485e-04],
           [-1.2184e+01, -1.2098e+01, -1.1176e+01,  ..., -1.6010e+01,
            -1.3081e+01, -1.0836e-04],
           ...,
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00]],

          [[-1.1193e+01, -1.0059e+01, -9.4249e+00,  ..., -1.4125e+01,
            -1.1683e+01, -7.6491e-04],
           [-1.1244e+01, -1.0294e+01, -9.5903e+00,  ..., -1.4202e+01,
            -1.1582e+01, -6.6842e-04],
           [-1.1528e+01, -1.0895e+01, -1.0285e+01,  ..., -1.5057e+01,
            -1.2287e+01, -3.6793e-04],
           ...,
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00]],

          [[-1.1499e+01, -1.1196e+01, -1.0036e+01,  ..., -1.4945e+01,
            -1.2375e+01, -2.9869e-04],
           [-1.1289e+01, -1.0855e+01, -9.9980e+00,  ..., -1.4556e+01,
            -1.2207e+01, -3.9927e-04],
           [-1.2123e+01, -1.2220e+01, -1.1154e+01,  ..., -1.5973e+01,
            -1.3356e+01, -9.6555e-05],
           ...,
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00],
           [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
            -3.9145e+00, -3.5911e+00]],
```

```
[[-1.1483e+01, -1.1163e+01, -1.0518e+01,  ..., -1.5118e+01,
  -1.2253e+01, -2.5388e-04],
 [-1.1310e+01, -1.1046e+01, -1.0525e+01,  ..., -1.5005e+01,
  -1.2228e+01, -2.8106e-04],
 [-1.1790e+01, -1.1803e+01, -1.0960e+01,  ..., -1.5847e+01,
  -1.2971e+01, -1.4054e-04],
 ...,
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00]],

[[-1.1245e+01, -1.0598e+01, -9.8038e+00,  ..., -1.4374e+01,
  -1.1770e+01, -5.0544e-04],
 [-1.1086e+01, -1.0426e+01, -9.6150e+00,  ..., -1.4132e+01,
  -1.1601e+01, -6.0433e-04],
 [-1.1303e+01, -1.1167e+01, -1.0117e+01,  ..., -1.4947e+01,
  -1.2256e+01, -3.1776e-04],
 ...,
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00]],

[[-1.1814e+01, -1.0889e+01, -1.0230e+01,  ..., -1.4779e+01,
  -1.2334e+01, -3.0263e-04],
 [-1.1797e+01, -1.0806e+01, -1.0395e+01,  ..., -1.4931e+01,
  -1.2494e+01, -3.0227e-04],
 [-1.2205e+01, -1.1848e+01, -1.1445e+01,  ..., -1.5643e+01,
  -1.3139e+01, -1.0514e-04],
 ...,
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00],
 [-1.7296e+00, -3.2166e+00, -3.9783e+00,  ..., -4.8099e+00,
  -3.9145e+00, -3.5911e+00]]])],
```

```
[tensor([[28, 28, 28,  ...,  0,  0,  0],
         [28, 28, 28,  ...,  0,  0,  0],
         [28, 28, 28,  ...,  0,  0,  0],
         [28, 28, 28,  ...,  0,  0,  0],
         [28, 28, 28,  ...,  0,  0,  0],
         [28, 28, 28,  ...,  0,  0,  0]])],
[tensor([[16, 12,  1, 25,  0,  2, 18,  5,  1, 11,  9, 14,  7,  0,  2,  1,  4,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
         [20, 21, 18, 14,  0, 21, 16,  0, 18,  9, 14,  7,  5, 18,  0, 22, 15, 12,
          21, 13,  5,  0,  2, 25,  0, 20,  5, 14,  0, 16,  5, 18, 19,  5, 14, 20,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
         [ 3,  1, 12, 12,  0, 10,  1,  3, 11,  0, 15, 14,  0,  8,  9, 19,  0, 21,
          11,  0, 14, 21, 13,  2,  5, 18,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
         [20, 21, 18, 14,  0, 15,  6,  6,  0, 13, 15, 14,  4,  1, 25,  0,  1, 12,
           1, 18, 13,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
         [ 3,  1, 12, 12,  0, 15, 14,  5,  0,  6, 15, 21, 18,  0, 20,  8, 18,  5,
           5,  0, 14,  9, 14,  5,  0,  6,  9, 22,  5,  0,  6, 15, 21, 18,  0,  6,
           9, 22,  5,  0,  6, 15, 21, 18,  0,  6, 15, 21, 18],
         [ 3,  1, 12, 12,  0,  1, 13,  5, 12,  9,  1,  0,  0,  0,  0,  0,  0,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
           0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0]])],
[tensor([17, 36, 26, 21, 49, 11])],
[tensor([220, 220, 242, 244, 308, 118])]]
```

ArgMax

In [26]:
```
greedy_hypotheses = post_process_predictions(evaluated_tensors[1], vocab)
```

In [27]:
```
print(greedy_hypotheses)
```

```
['play breaking bad ', 'turn up ringa volume by ten percent ', 'he called jack on his uk number ', 'turn off
monday alarm ', 'call one four three nine five four five four four ', 'call amelia ']
```

In [29]:
```
references = post_process_transcripts(evaluated_tensors[2], evaluated_tensors[3], vocab)
```

In [30]:
```
references
```

Out[30]:
```
['play breaking bad',
 'turn up ringer volume by ten persent',
 'call jack on his uk number',
 'turn off monday alarm',
 'call one four three nine five four five four four',
 'call amelia']
```

In [31]:
```
wer = word_error_rate(hypotheses=greedy_hypotheses, references=references)
logging.info("Greedy WER {:.2f}%".format(wer * 100))
```

```
[NeMo I 2020-05-19 18:37:18 <ipython-input-31-6f34eec6e517>:2] Greedy WER 12.50%
```

$$\text{Word Error Rate} = \frac{\text{Insertions} + \text{Deletions} + \text{Substitutions}}{\text{Number of Words in Reference Transcript}}$$

Insertions The quick slick brown fox

Deletions The _____ brown fox

Substitutions The slick brown fox

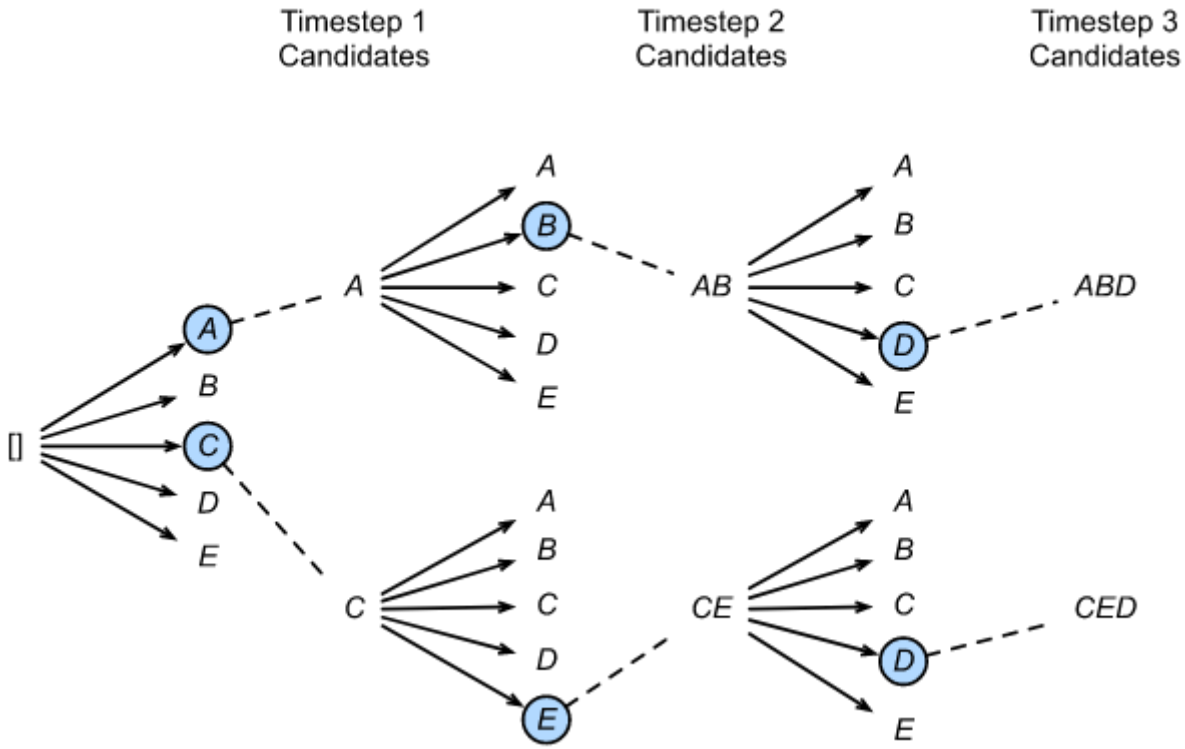
In [32]:
```
# Convert logits to list of numpy arrays
logprob = []
for i, batch in enumerate(evaluated_tensors[0]):
    for j in range(batch.shape[0]):
        logprob.append(batch[j][: evaluated_tensors[4][i][j], :].cpu().numpy())
```

In [33]:
```
logprob[0][30]
```

Out[33]:
```
array([-1.2477631e+01, -1.4158891e+01, -1.4817966e+01, -1.6021255e+01,
       -1.6142181e+01, -1.4927215e+01, -1.5493126e+01, -1.5764254e+01,
       -1.4164321e+01, -1.4987330e+01, -1.7241220e+01, -1.5790519e+01,
       -1.4521273e+01, -1.4006466e+01, -1.4994835e+01, -1.4535931e+01,
       -1.5819631e+01, -1.7521524e+01, -1.5811963e+01, -1.7527599e+01,
       -1.4031932e+01, -1.4865693e+01, -1.6257439e+01, -1.5289449e+01,
       -1.7705647e+01, -1.6363016e+01, -1.8240097e+01, -1.4467798e+01,
       -1.1563234e-05], dtype=float32)
```

In [34]:
```
beam_wers = []
logprobexp = [np.exp(p) for p in logprob]
```

BeamSearch


Timestep 1
Candidates
Timestep 2
Candidates
Timestep 3
Candidates
[]
A
B
C
D
E
A
B
C
D
E
AB
ABD
CE
CED

In [47]:
```
alpha = 1.12
beta = 0.12
beam_search_with_lm = nemo_asr.BeamSearchDecoderWithLM(
            vocab=vocab,
            beam_width=100,
            alpha=alpha,
            beta=beta,
            lm_path='/home/artsokol/wer_1643/lm/5gramm_kenlm.binary',
            num_cpus=max(os.cpu_count(), 1),
            input_tensor=False,
        )
```

In [48]:
```
beam_predictions = beam_search_with_lm(log_probs=logprobexp, log_probs_length=None, force_pt=True)

beam_predictions = [b[0][1] for b in beam_predictions[0]]
```

In [50]:
```
beam_predictions
```

Out[50]:
```
['play breaking bad',
 'turn up ringtones by ten percent',
 'call jack on his uk number',
 'turn off monday alarm',
 'call on for three nine five four five four four',
 'call amelia']
```

In [51]:
```
references
```

Out[51]:
```
['play breaking bad',
 'turn up ringer volume by ten persent',
 'call jack on his uk number',
 'turn off monday alarm',
 'call one four three nine five four five four four',
 'call amelia']
```

In [52]:
```
lm_wer = word_error_rate(hypotheses=beam_predictions, references=references)
```

In [53]:
```
logging.info("Beam WER {:.2f}%".format(lm_wer * 100))
```

```
[NeMo I 2020-05-19 18:55:21 <ipython-input-53-807c8dc63473>:1] Beam WER 15.62%
```

['play breaking bad ', 'turn up ringa volume by ten percent ', 'hecalled jack on his uk number ', 'turn off monday alarm ', 'call one four three nine five four five four four ', 'call amelia ']

In [ ]:
```

```