Disclaimer: This presentation is a work of fiction written from the perspective of a 2020 researcher traveling back in time to mid 2013 to share some 2020 xNN based application ideas; references to credit the actual inventors of the various ideas is provided at the end

DeepSpeech2: SoundWaves to Characters

Nancy Dominguez
nsd093020@utdallas.edu
May 1, 2013

Disclaimer: This presentation is a work of fiction written from the perspective of a 2020 researcher traveling back in time to mid 2013 to share some 2020 xNN based application ideas; references to credit the actual inventors of the various ideas is provided at the end

Motivation

- Many professions and situations require note-taking
  - Doctors require scribes
  - Courts require stenographers
- Improve access for people with
  - Physical disabilities
  - Learning disabilities
  - Blindness
- Future Applications
  - Voice enabled cars
  - Create more sophisticated voicebots

1784

Wolfgang von Kempelen creates the **Acoustic-Mechanical Speech Machine** in Vienna

Thomas Edison invents the first dictation machine

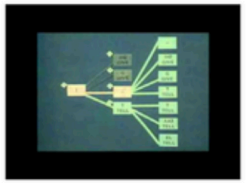
1879

1952

Bell Labs releases **Audrey**, capable of recognizing spoken digits with **90% accuracy** - but only when spoken by its inventor

IBM Shoebox can understand 16 English words

1962



1971

Harpy, created at Carnegie Mellon University, can comprehend **1,011 words** - and some phrases

IBM Tangora, using the Hidden Markov Model, predicts upcoming phonemes in speech

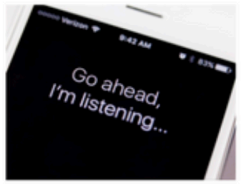
1986

2006

The National Security Agency (NSA) starts using speech recognition to isolate key words in recorded speech

Google launches a voice search app, bringing speech recognition to mobile devices

2008



2011

Apple announces Siri, ushering in the age of the voice-enabled digital assistant

Figure 1: [“Speech Recognition Technology: The Past, Present, and Future.”]

ClickZ

Key Insight

- Currently the most popular methods have included Hidden Markov Models
- HMM issues for speech to text application
  - Goal: Soundwaves to phonemes
    - Requires a pronunciation model
  - Markov assumption does not always hold
    - probability of being in a given state at time t only depends on the state at time t-1
  - Amount of data required to train an HMM is very large
  - Requires soundwaves to be transformed to mel frequency cepstral coefficients (MFCCs)
    - This transformation results in information loss

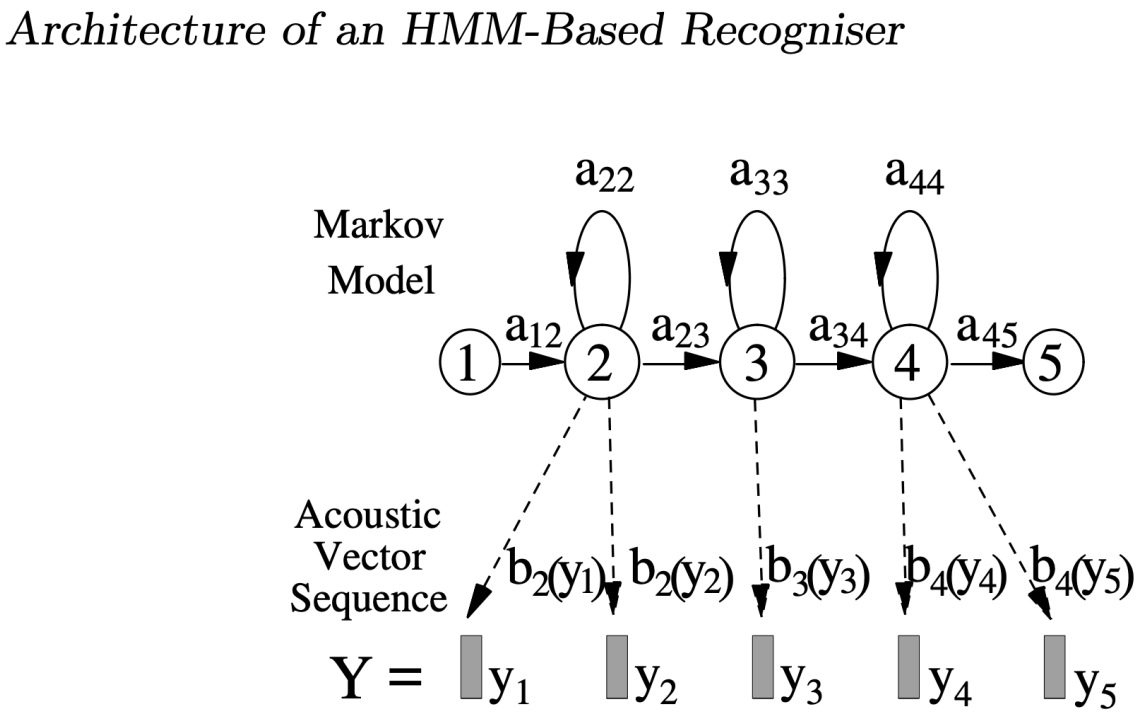


Figure 2: "The Application of Hidden Markov Models in Speech Recognition"

Key Insight: Basic Overview

- Goal: soundwaves to graphemes (lexical characters)
- Approach: Match segments of time of a sound clip with lexical characters.
  - x: Spectrogram (sound input sentence) with time domain t and power frequency p, i.e. spectrogram
  - l: classes of possible lexical characters

    {a, …, z, _}
  - $p(l_t|x)$: Given input $\mathbf{x_{t,p}}$, which class found within l has the highest probability at time t?
  - i.e. each time step has a power frequency that maps to a lexical character

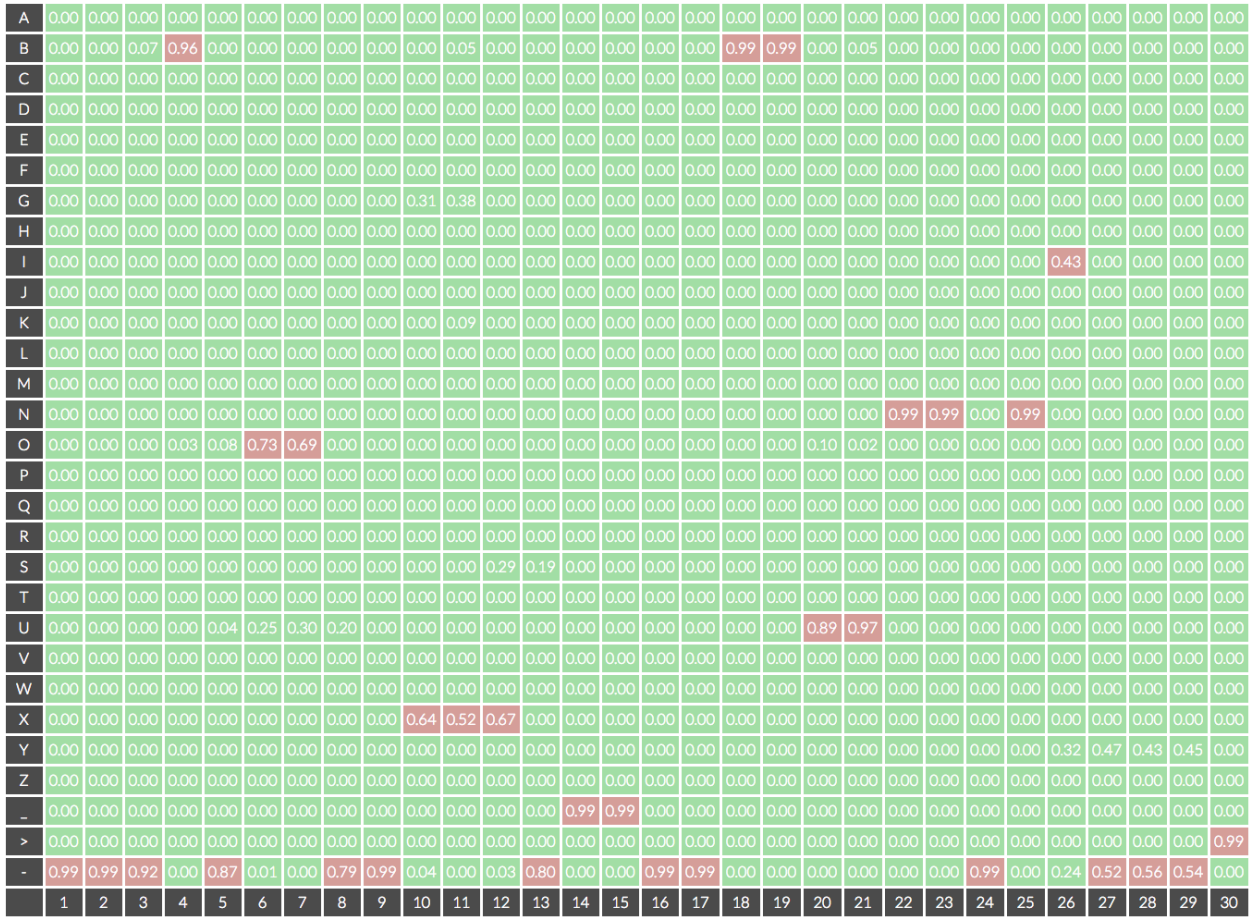


Figure 3: "CTC Networks and Language Models: Prefix Beam Search Explained"

Key Insight: Data Preprocessing

- Transform sound waves to Mel Spectrograms and normalize
  - Mel Spectrum Scale aims to mimic the human ear perception of sound
    - is more discriminative at lower frequencies and less discriminative at higher frequencies
  - Can be seen as “images” of sound
- Unlike HMMs, DeepSpeech2 is not susceptible to correlated input (Mel Spectograms)
  - Does not require MFCCs

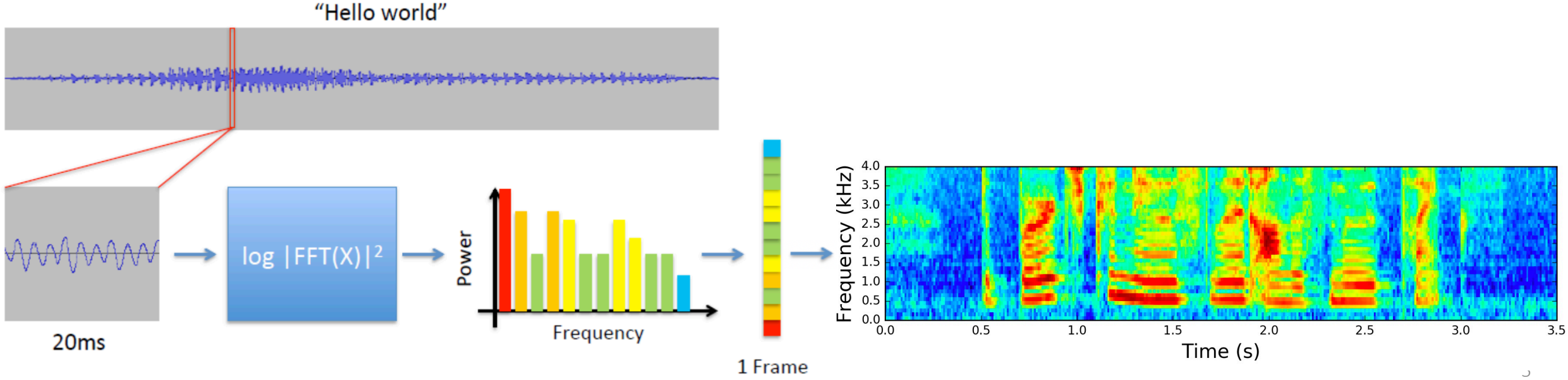
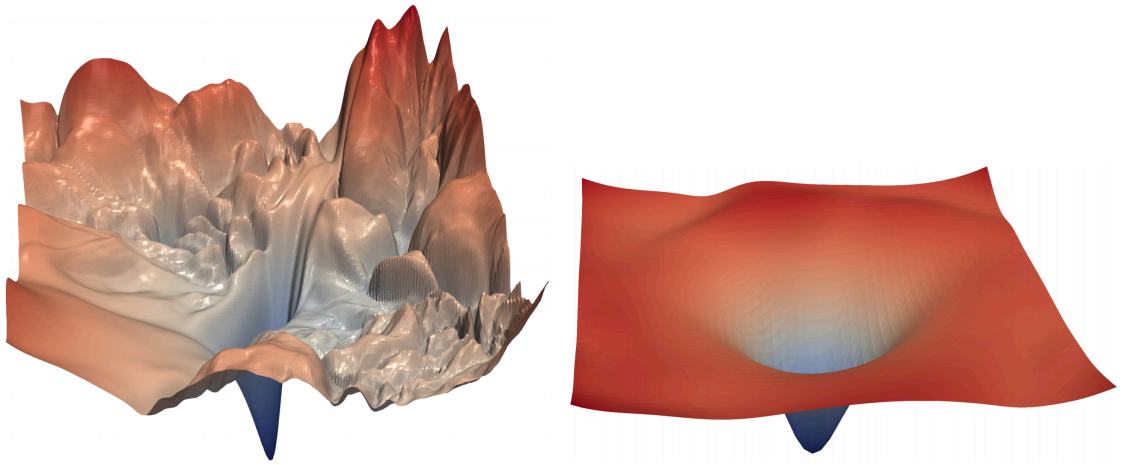


Figure 4: "Speech Processing for Machine Learning" and "Speech Recognition"

Key Insight: Data Preprocessing

- Recommended for model optimization
- SortaGrad
  - Idea: presenting examples in order of difficulty can accelerate online learning
  - Length of utterance is treated as a heuristic or difficulty
    - longer utterances considered more difficult since they have a higher CTC function cost
  - During first epoch, SortaGrad groups utterances with similar lengths together, in decreasing order
- BatchNorm and residual connections
- Improves model stability, generalizability, and results in faster convergence



(a) without skip connections (b) with skip connections

Figure 5: "Visualizing the Loss Landscape of Neural Nets"

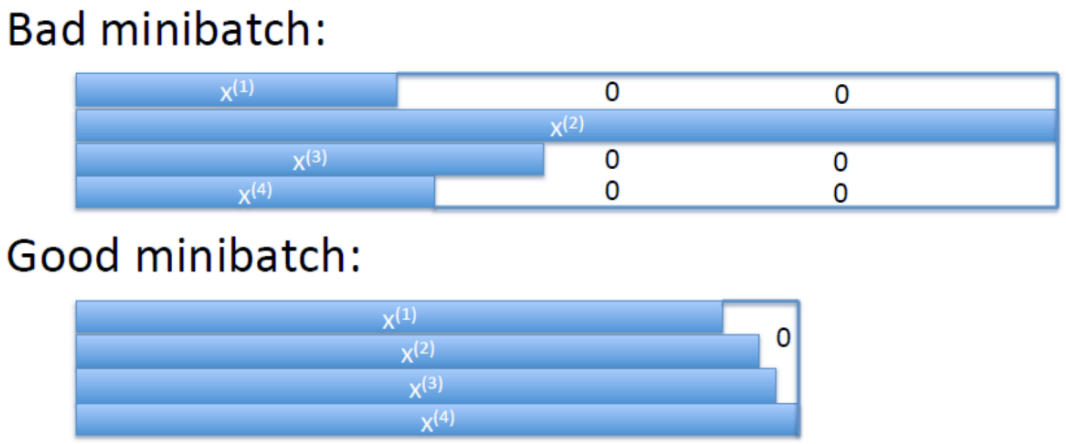


Figure 6: "Speech Recognition"

Key Insight: Data Augmentation

- If data is scarce , Spectrogram Augmentation (SpecAugment) is recommended to improve accuracy
  - Blocks out, random frequencies channels and time steps
  - Authors achieved 6.8% WER on LibriSpeech 960h without use of Language Model

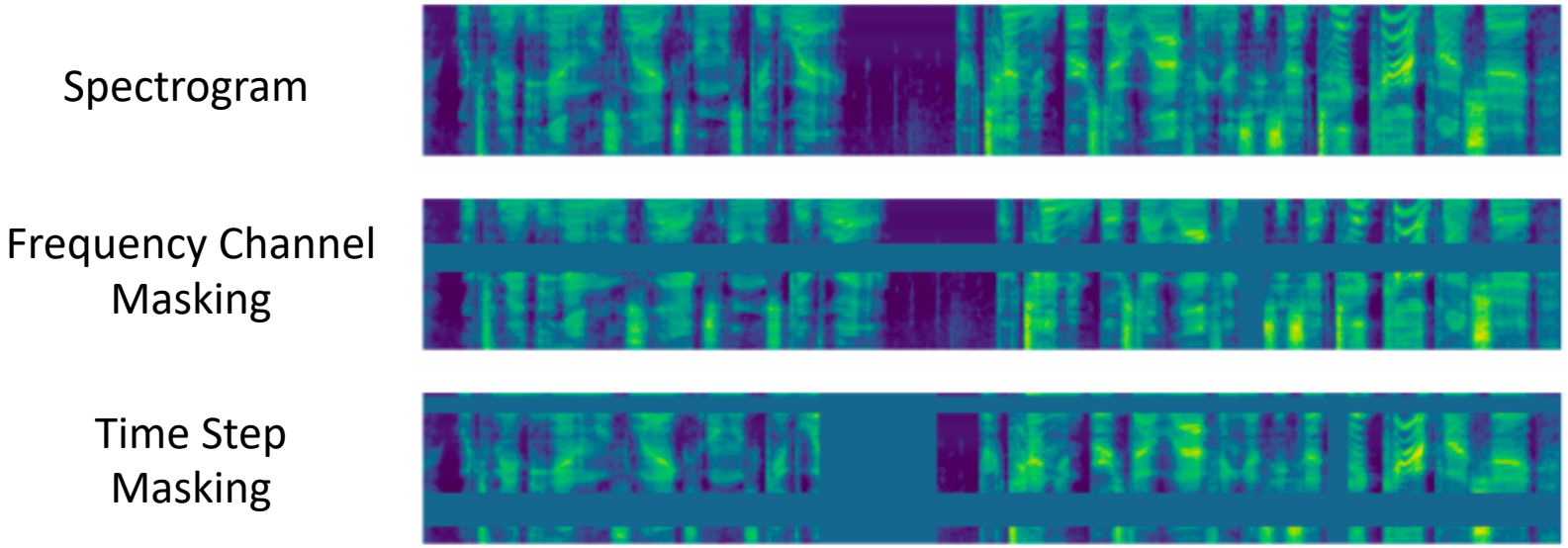


Figure 7: "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition"

Proposed Approach

- 1-3 2D convolution layers
  - Leverage their feature extraction power to shorten the number of time-steps required at the RNN layers
  - Convolutional Layers can reduce spectral variation resulting in improved WER
- 1-7 bidirectional RNN-style layers
  - Converts an input speech sequence to a final text transcription
  - GRU cells are chosen as they are more accurate than a vanilla RNN and are faster to train and less likely to diverge than LSTM cells for smaller datasets
    - GRU and LSTM are found to be comparable for larger datasets
- Increase depth of model as datasets grow for better learning

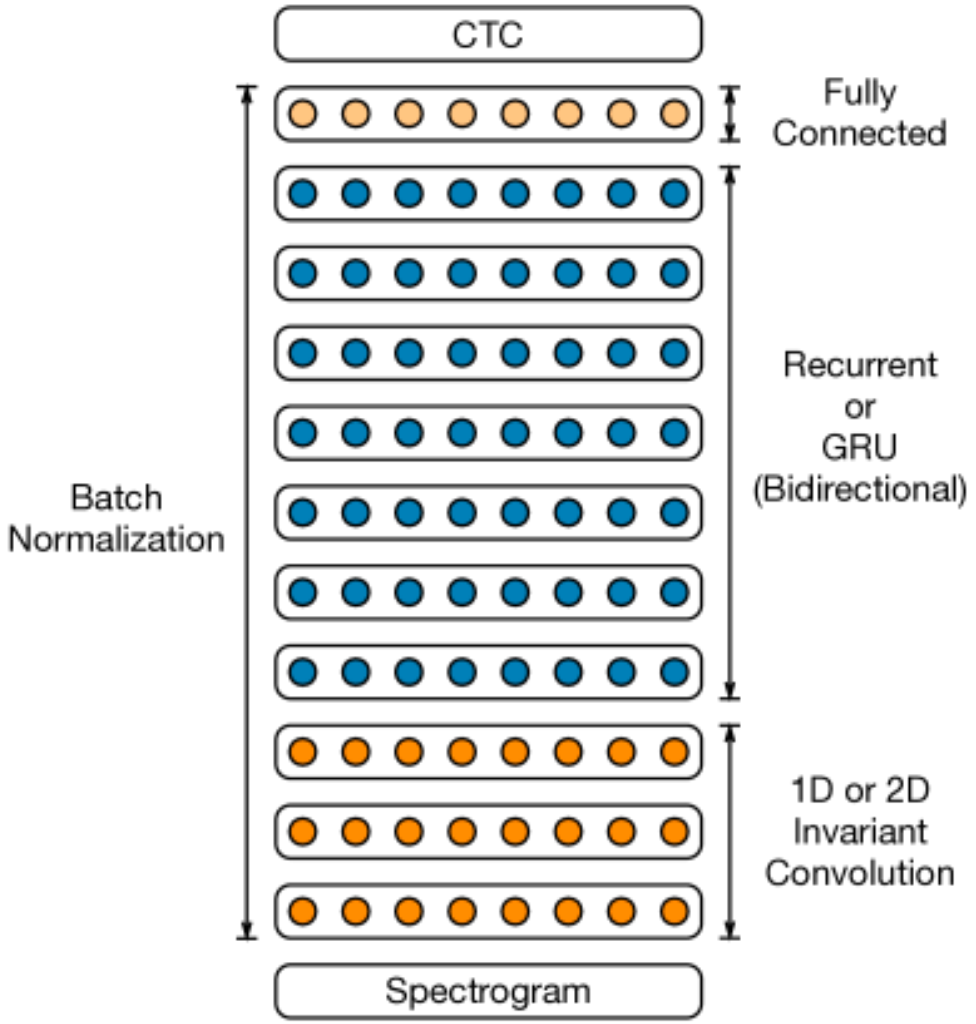


Figure 8: "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin"

Proposed Approach: Closer Look

- At each timestep t of the bidirectional RNN a corresponding Spectrogram frame is given
  - looks at the previous $t-1$ and future $t+1$ timesteps to output a probability vector of all lexical graphemes
- We're given sound input and its text transcription, but we don't know the corresponding alignment between the Spectrogram frame and the grapheme
  - Use CTC loss function to handle this

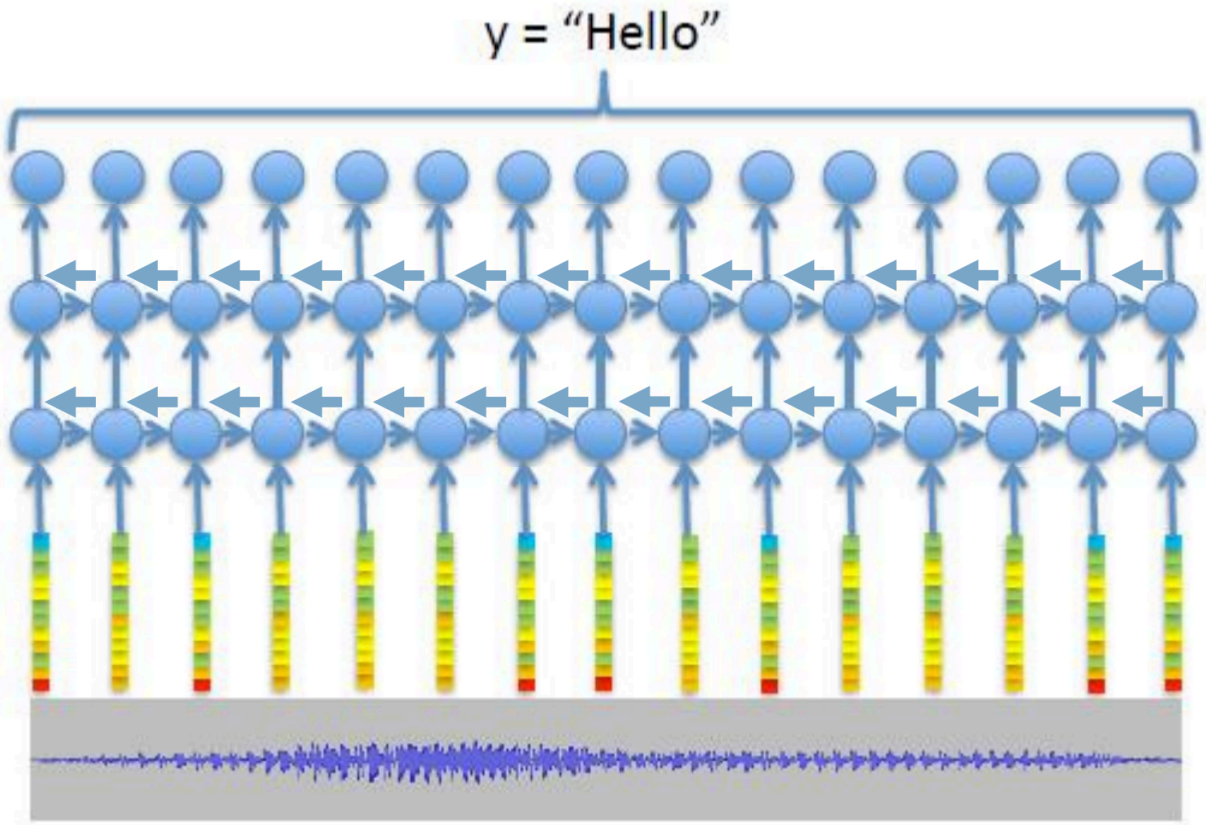


Figure 9: "Speech Recognition"

Proposed Approach: Decoder

- CTC: Connectionist Temporal Classification
  - Alignment-free: does not need to know the alignment between the input and output
- At each timestep, the RNN layer outputs a probability vector of {h,e,l,o,_}
  - Darkest is most probable
- CTC then computes the probability of possible sequences
  - Darkest is most probable
- To generate output:
  - Merge repeated characters
  - Remove spaces

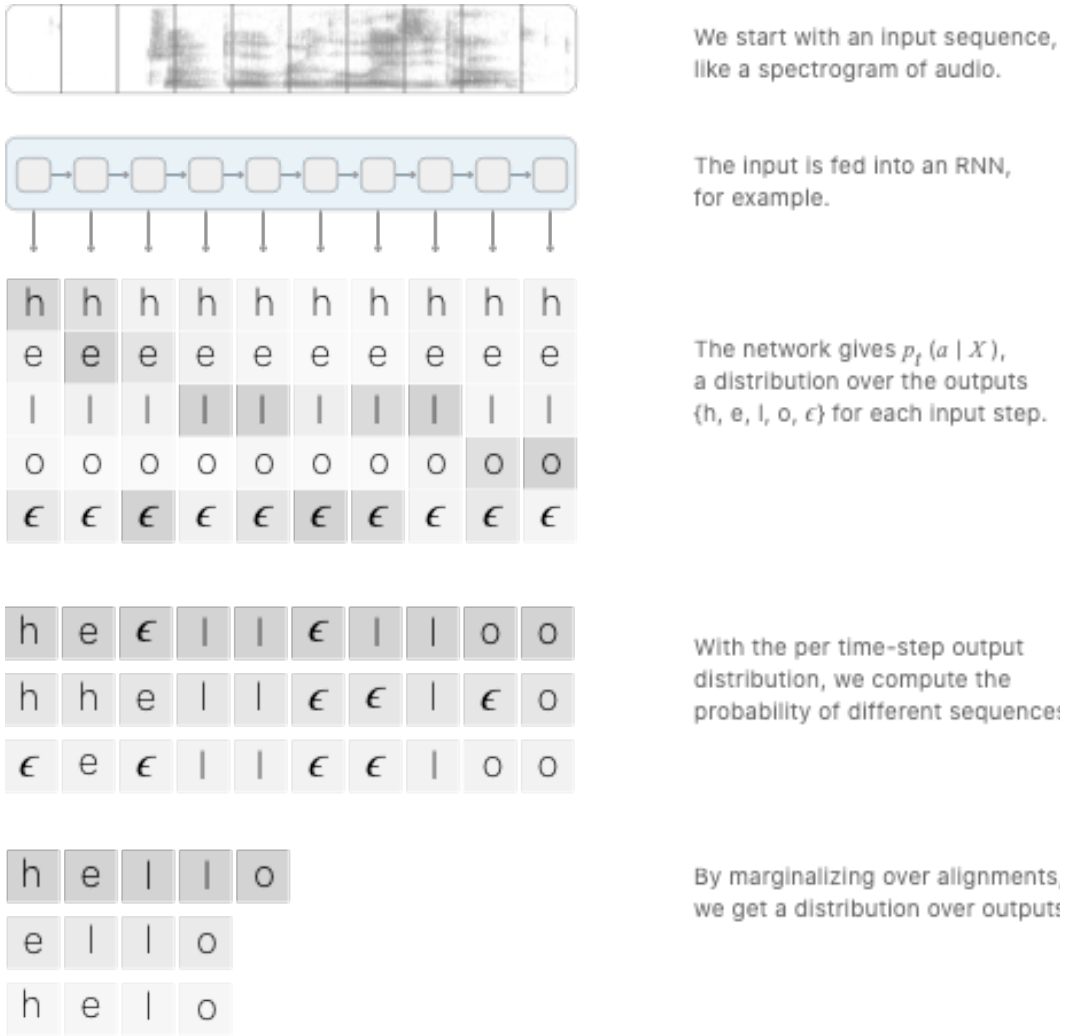


Figure 10: "Sequence Modeling With CTC"

Proposed Approach: Decoder

- To compute the probability of each CTC output:
- $P(output|input)$ =sum of all unmerged sequences that produce the same output

- **Training**:
- Network Parameters are optimized to maximize the likelihood of the correct label
- **Testing**:
- Greedy decoding is used to select the output with the highest probability

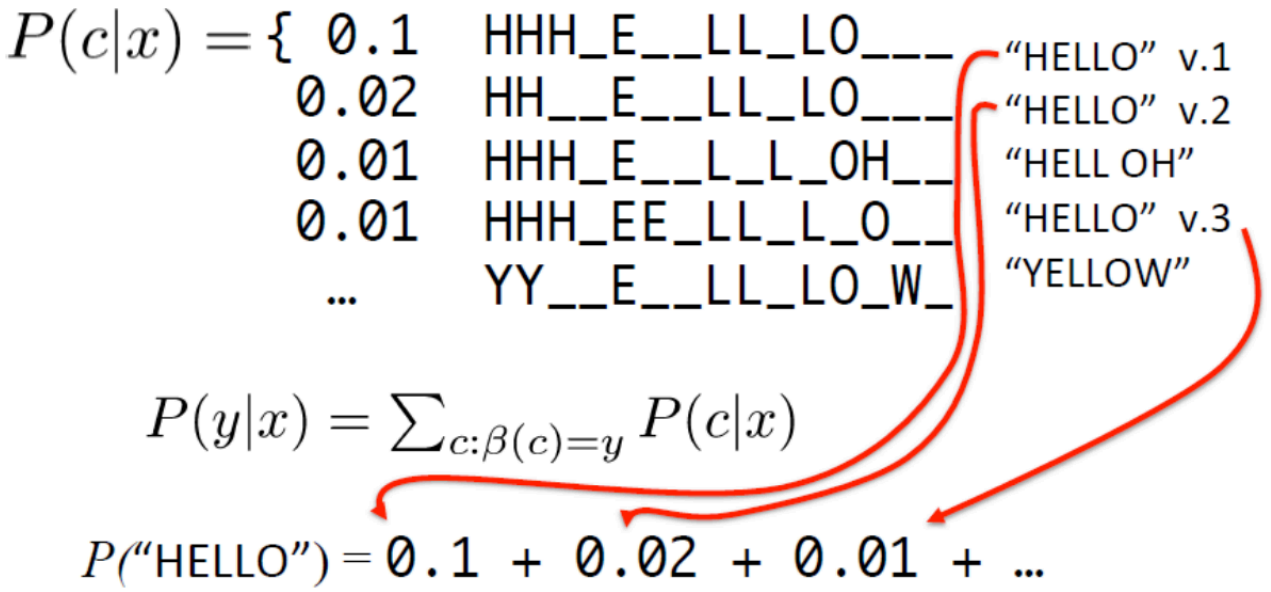


Figure 11: "Speech Recognition"

Results

Code: https://colab.research.google.com/drive/1qqHHzdseMqMYl61VN4ohaB2qa6EpyXDN
Results: https://www.comet.ml/yaygreat/deepspeech2/view/new

- All networks tested have 3 Conv2D layers with either 1, 5, or 7 bidirectional GRU layers and varying batch sizes of 5, 10, and 20 samples
  - Batch size = 32 required more memory than allowable on Google Colab
- LibriSpeech-100 was used to train and test
  - 100 hours of speech derived from read audiobooks from the LibriVox project
  - LibriSpeech-360 required more memory than allowable on Google Colab
- Metrics
- **WER- Word Error Rate**
  - Measures the difference, using levenshtein distance, between the hypothesis and the true label at the word level
- **CER- Character Error Rate**
  - Measures the difference, using levenshtein distance, between the hypothesis and the true label at the character level
- All figures were collected using Comet ML API

Results

- All experiments ran for 10 epochs
  - Larger networks could run for longer to obtain better accuracy
- The most accurate parameters for the dataset type and size used is one with 5 biGRU and 3 Conv2D layers with batch size of 5
  - Smaller batch sizes converged faster for each network
  - Smaller networks also converged faster
- At the final epoch, the 5biGRU layer network outperformed both 1 biGRU and 7 biGRU layer networks in accuracy (WER, CER)

CER and WER are average across all test samples

	BATCH SIZE					
NUMBER OF BIGRU LAYERS	5	10	5	10	5	10
	FINAL AVG WER	FINAL AVG CER	FINAL AVG WER	FINAL AVG CER	FINAL AVG WER	FINAL AVG CER
1	0.4843	0.1638	0.5026	0.1707	0.5428	0.1885
5	0.4370	0.1454	0.4549	0.1527	0.5697	0.1966
7	-	-	0.4840	0.1636	-	-

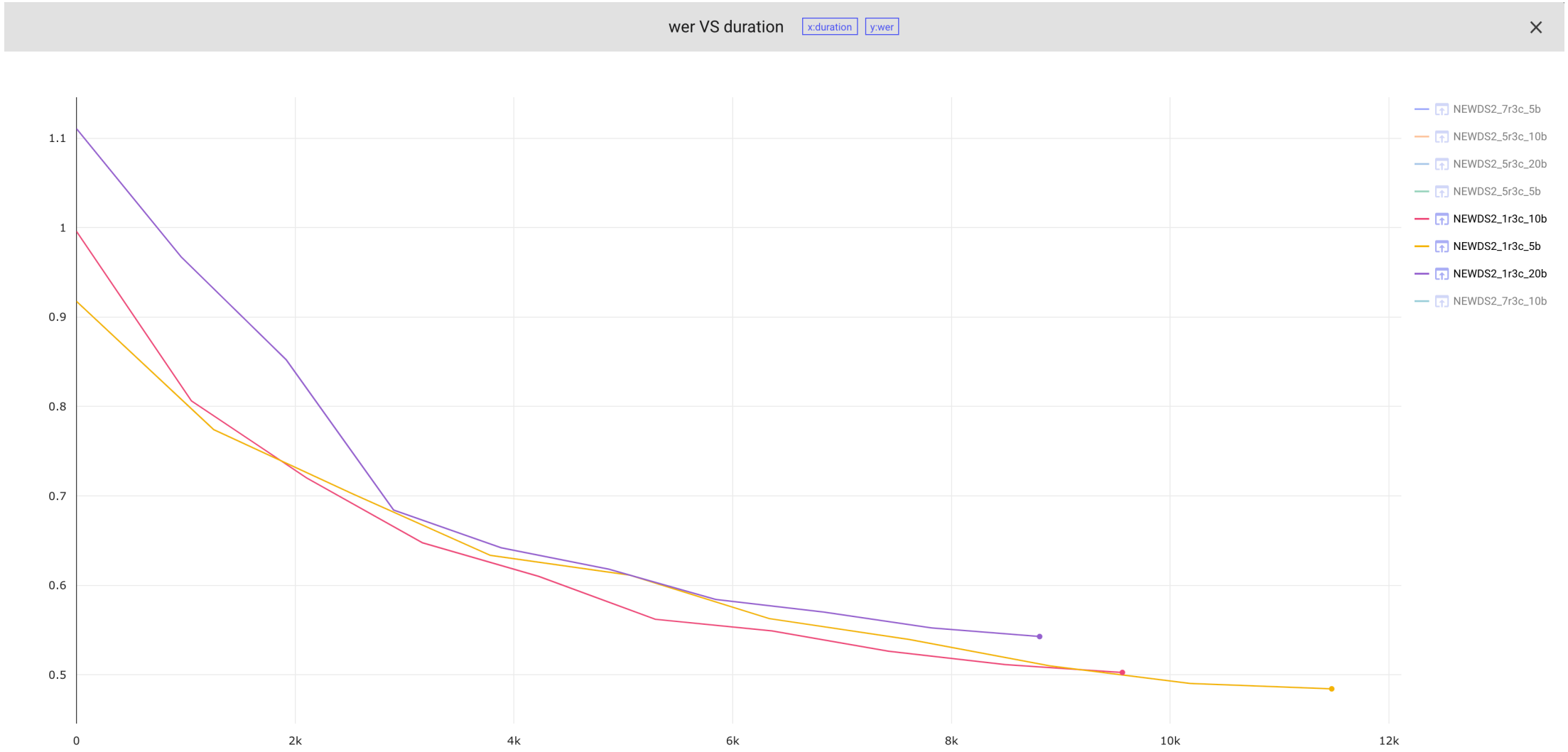
Sample utterance

6930-76324-0020.flac: YET LITTLE AS IT WAS IT HAD ALREADY MADE A VAST DIFFERENCE IN THE ASPECT OF THE ROOM

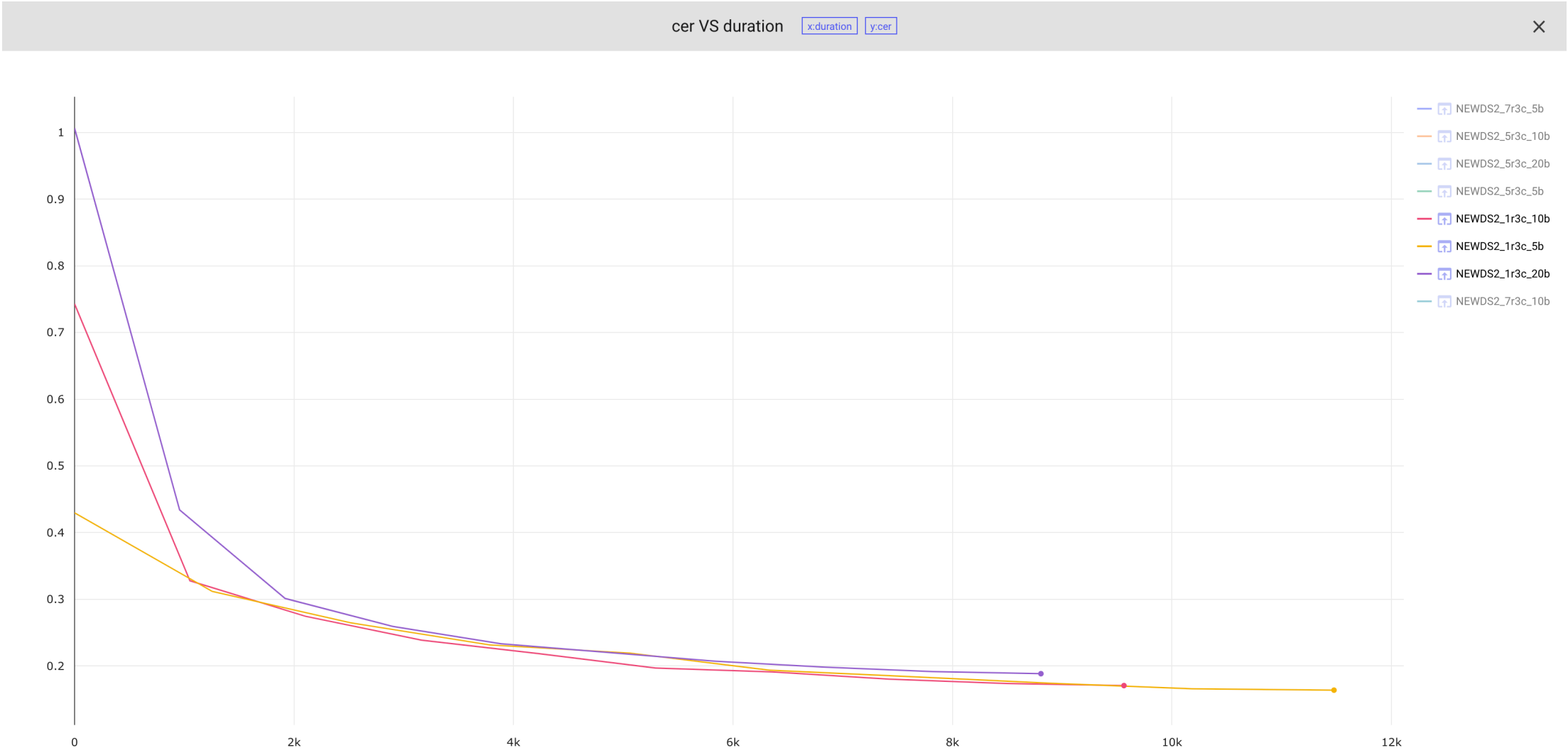
5r3c_5b: yet little as it was it had arereay made a fast difference in the aspect up the room

1r3c_5b: yet little as it was it had ar reany made of fast difference in the aspect of the ro

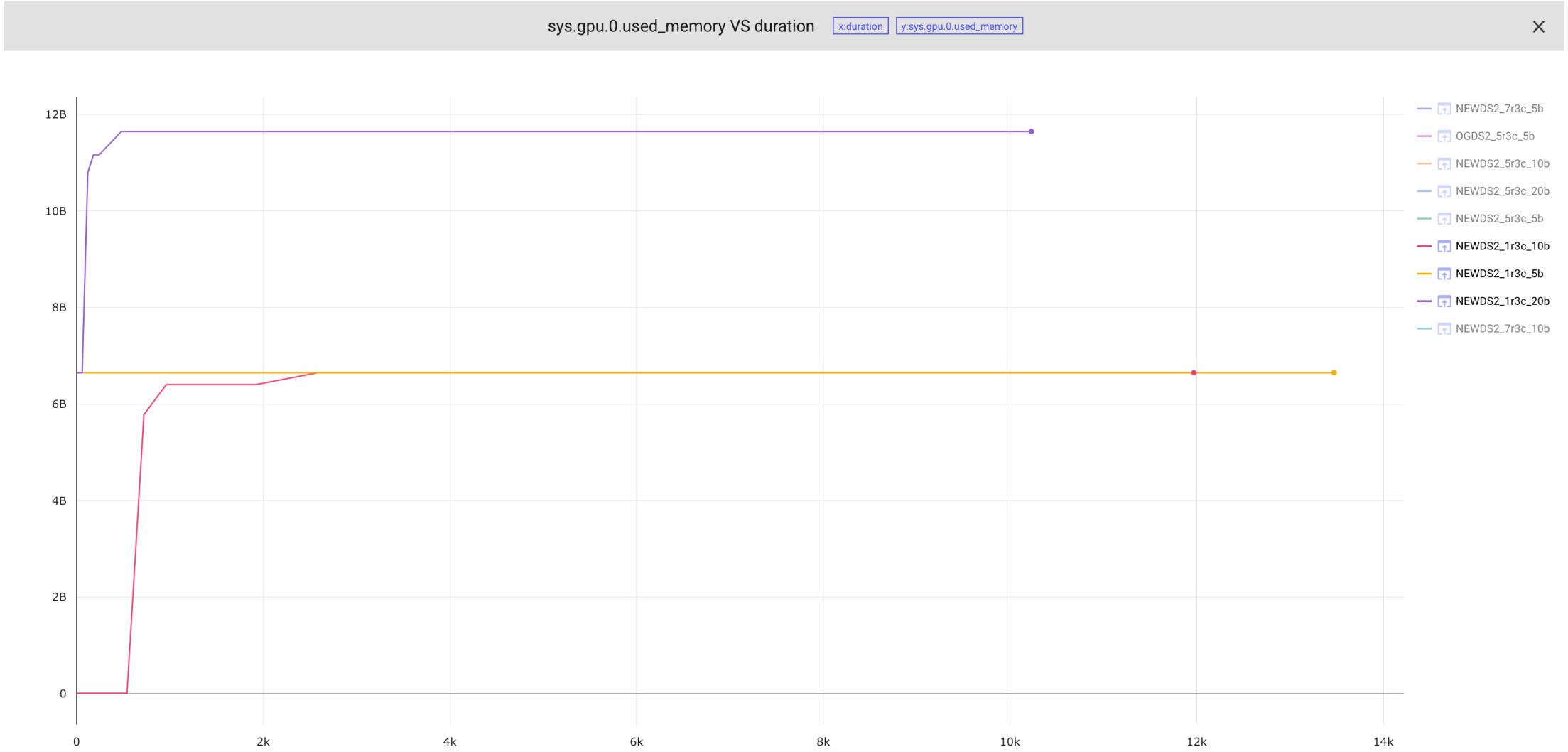
Results: 1 bidirectional GRU layer



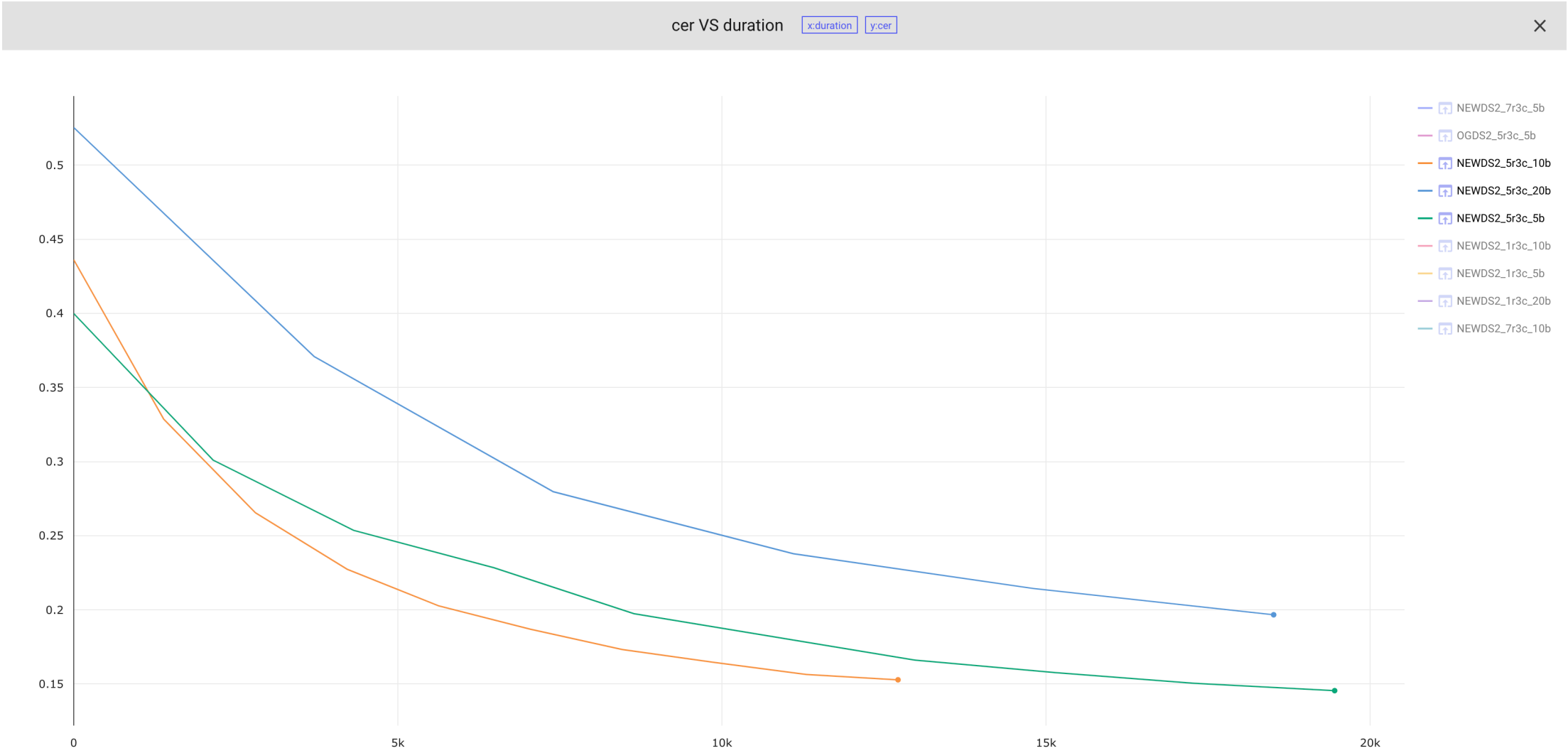
Results: 1 bidirectional GRU layer



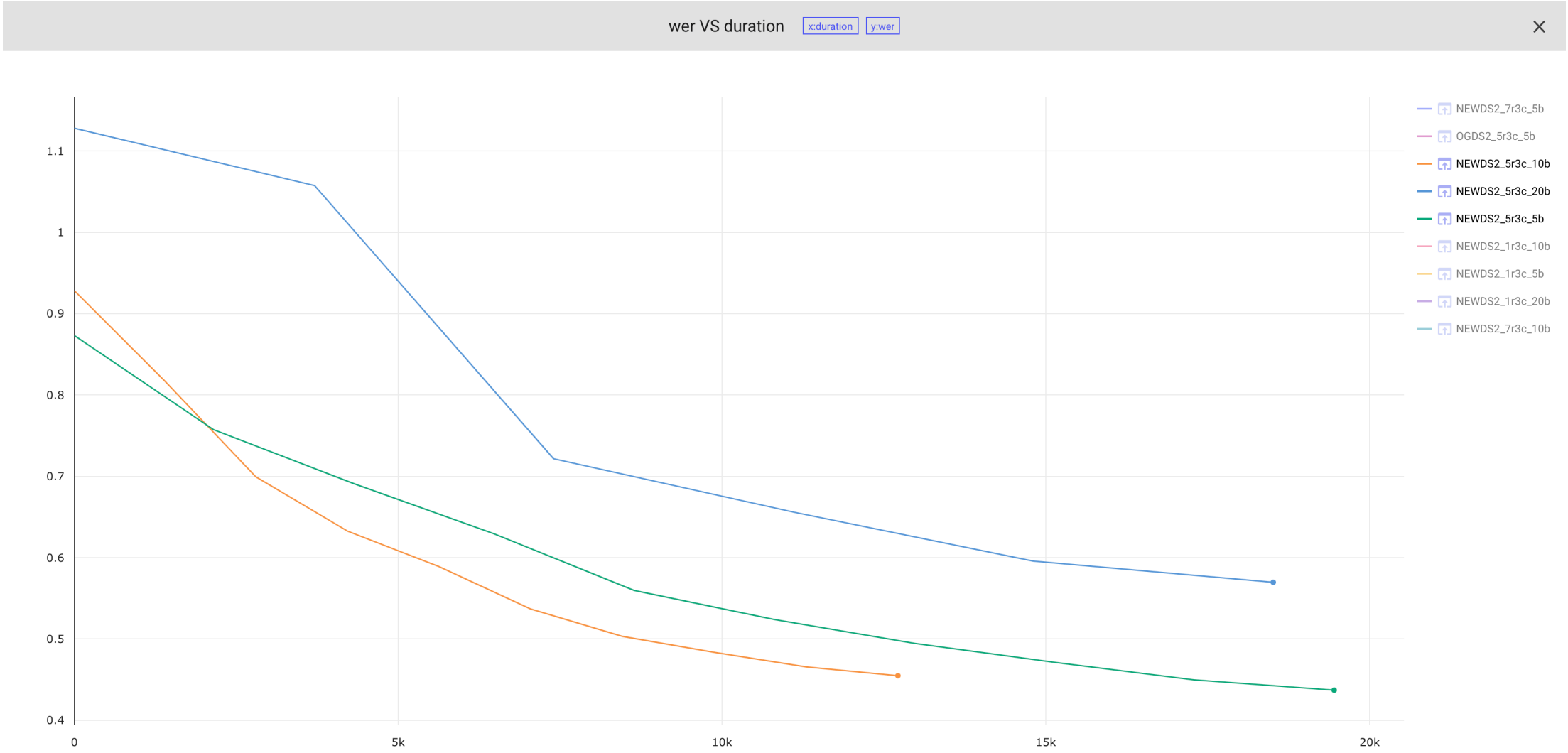
Results: 1 bidirectional GRU layer



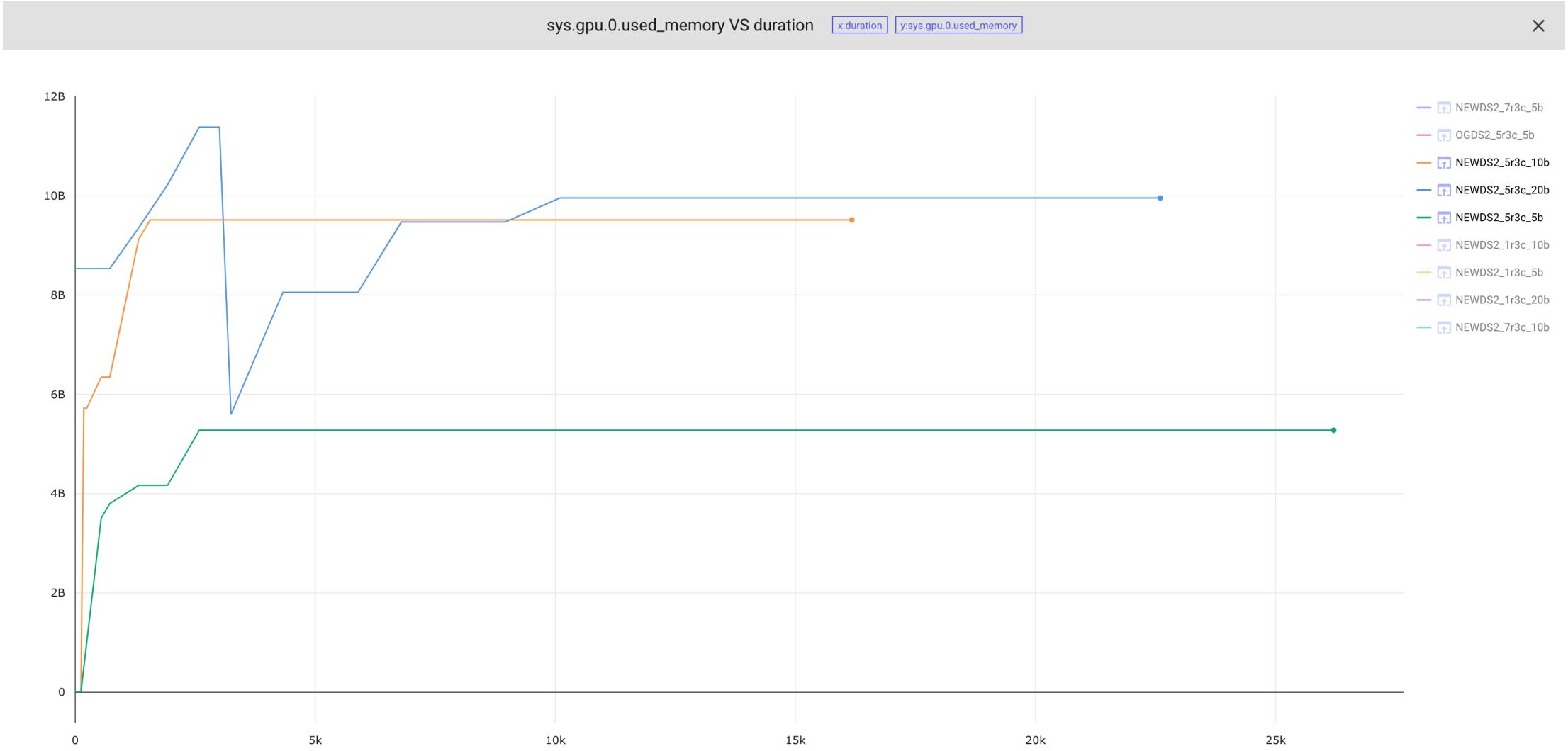
Results: 5 bidirectional GRU layers



Results: 5 bidirectional GRU layers

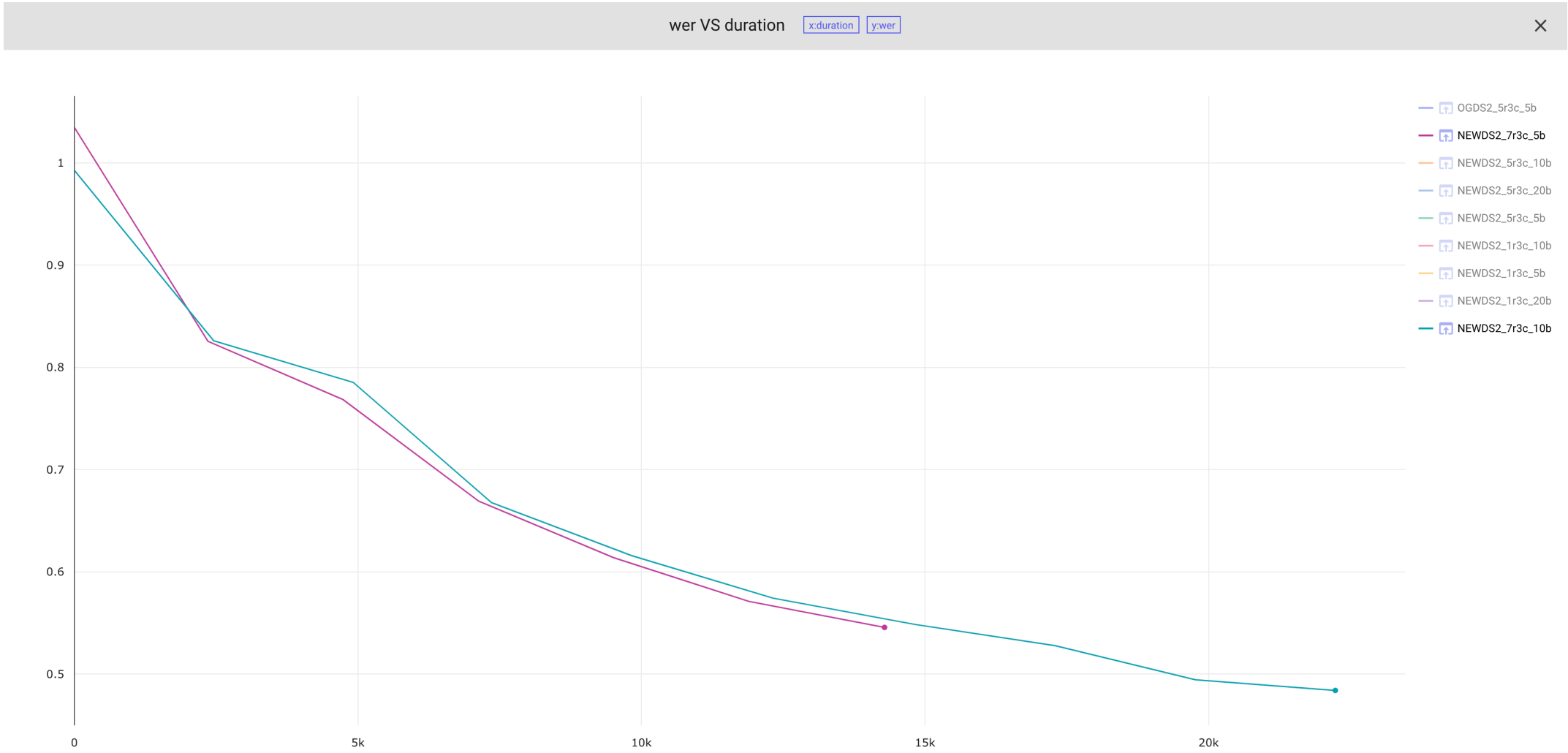


Results: 5 bidirectional GRU layers



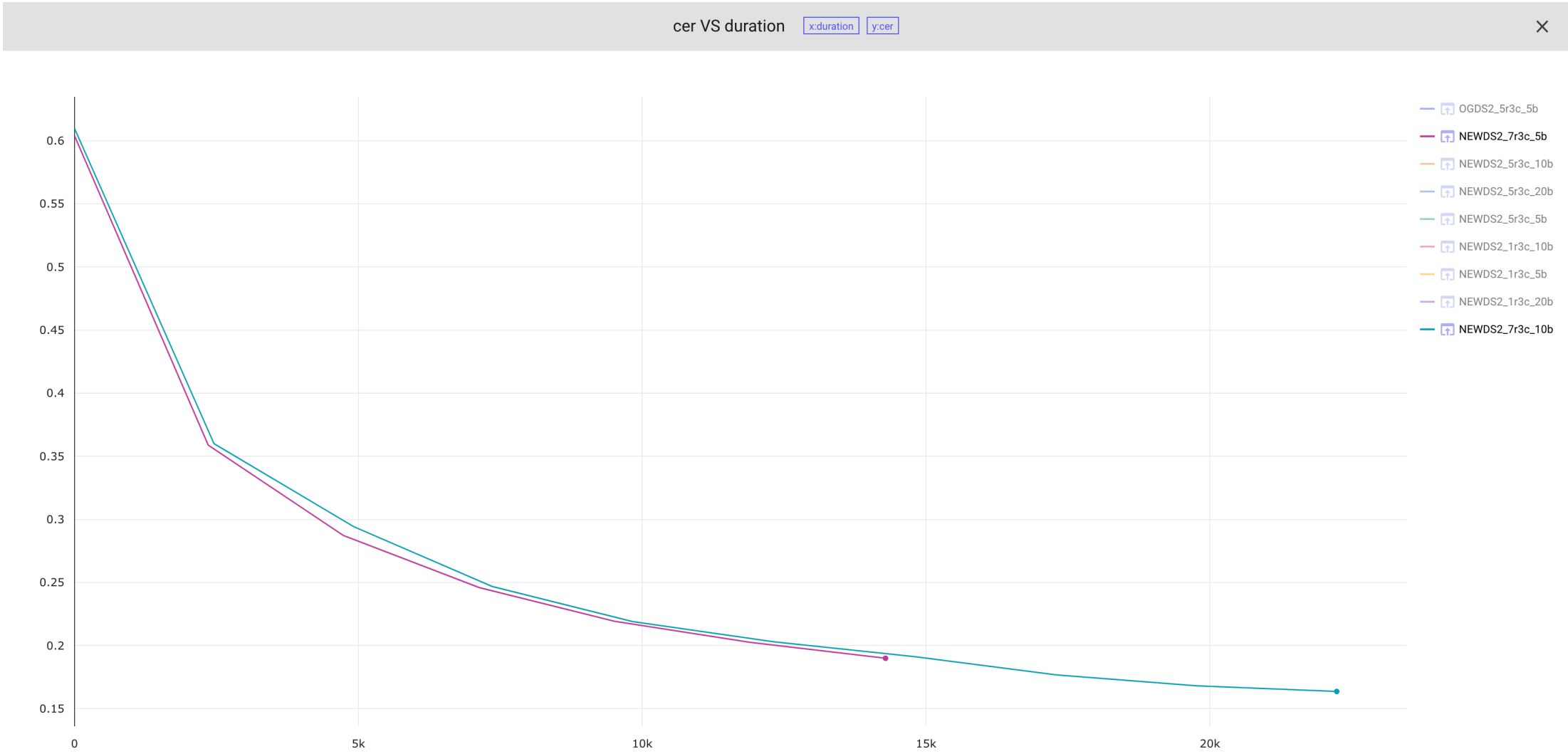
Results: 7 bidirectional GRU layers

Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



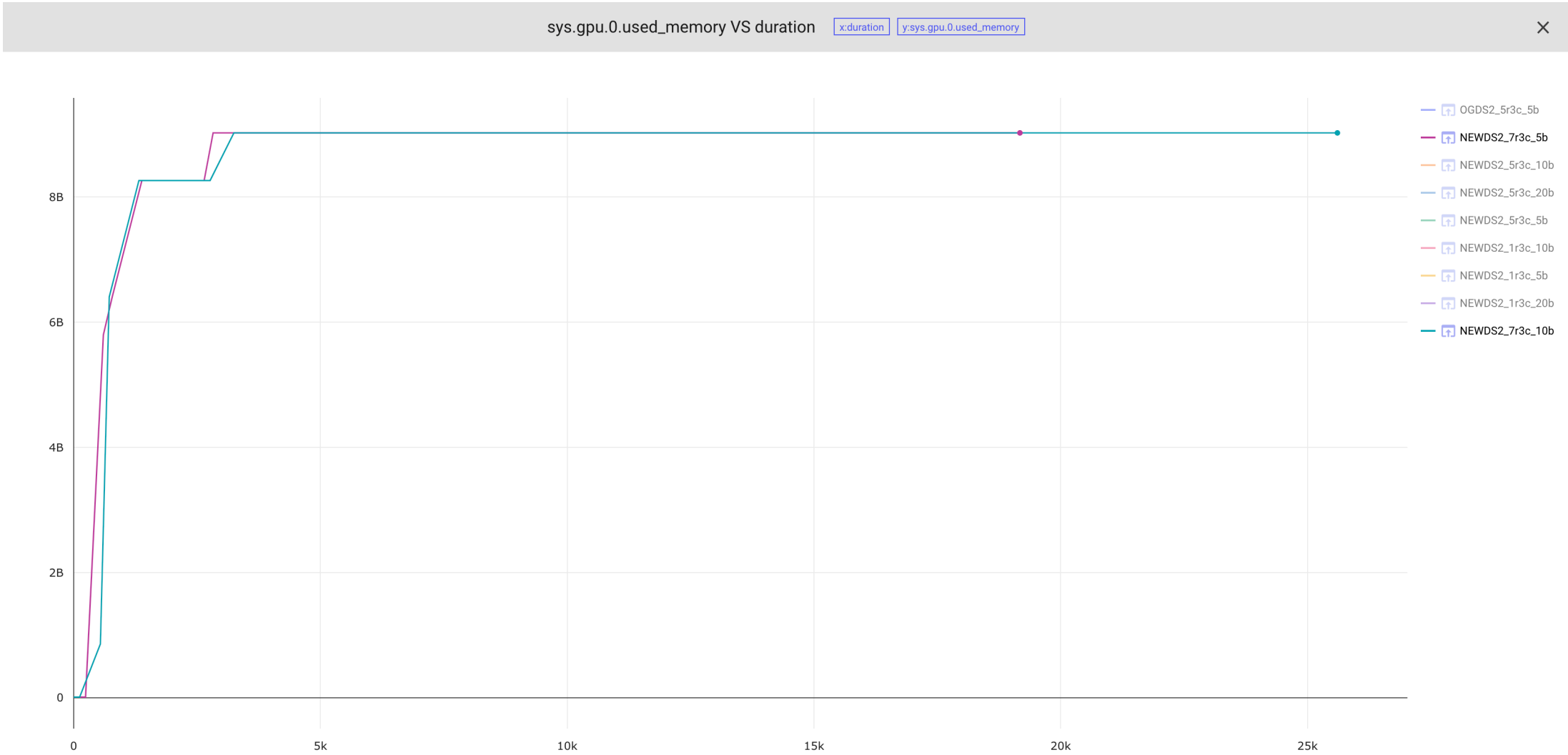
Results: 7 bidirectional GRU layers

Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



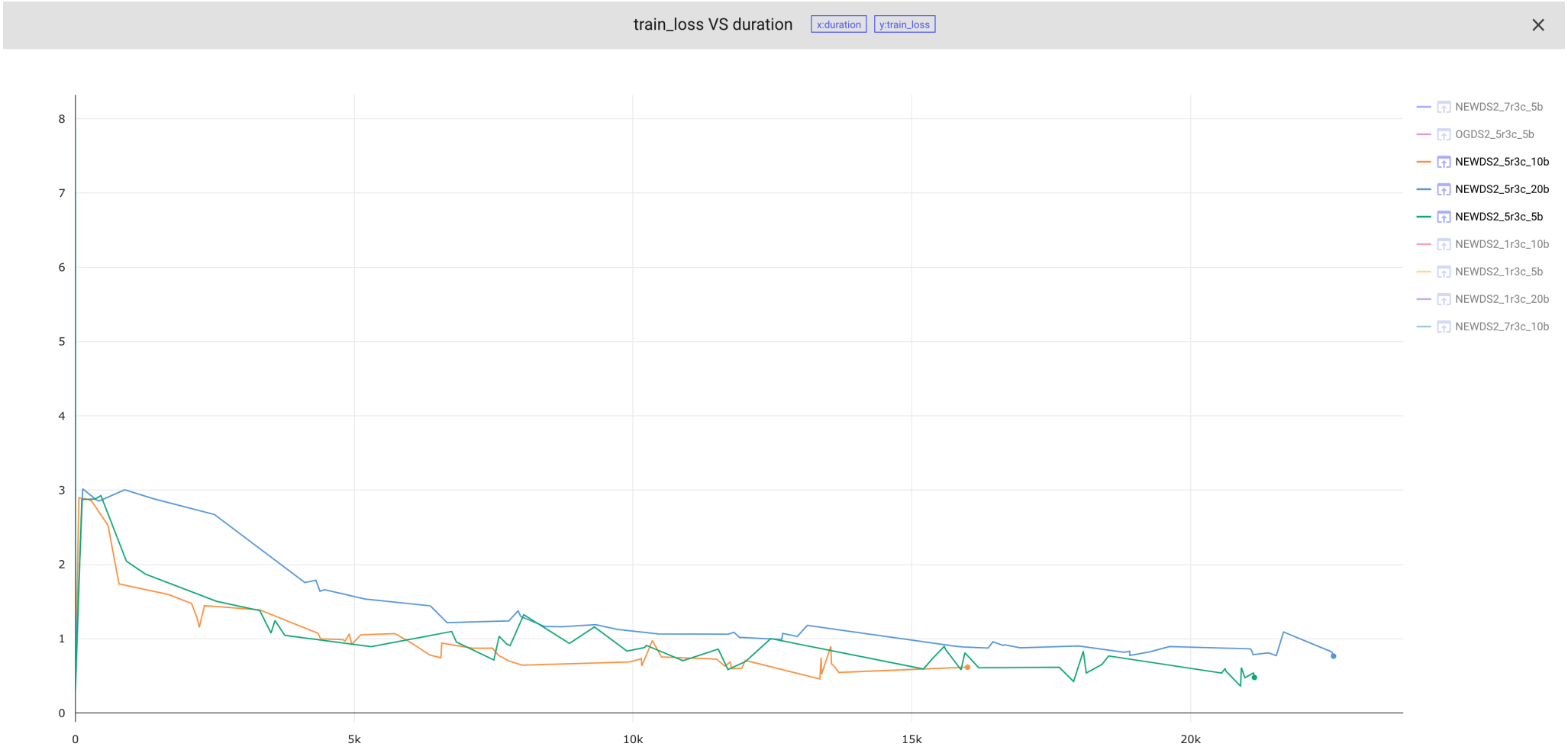
Results: 7 bidirectional GRU layers

Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



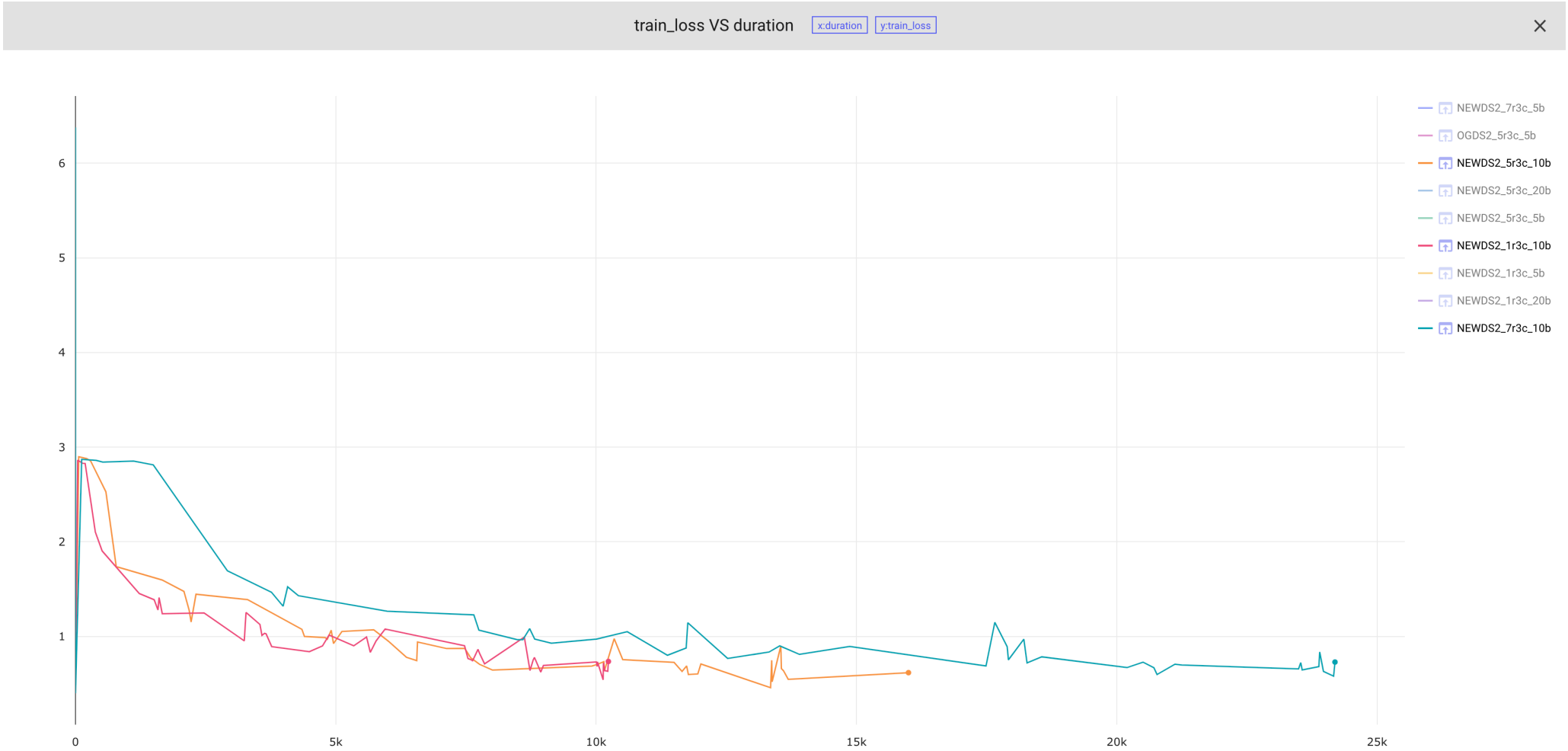
Results

- Smaller batches converge faster



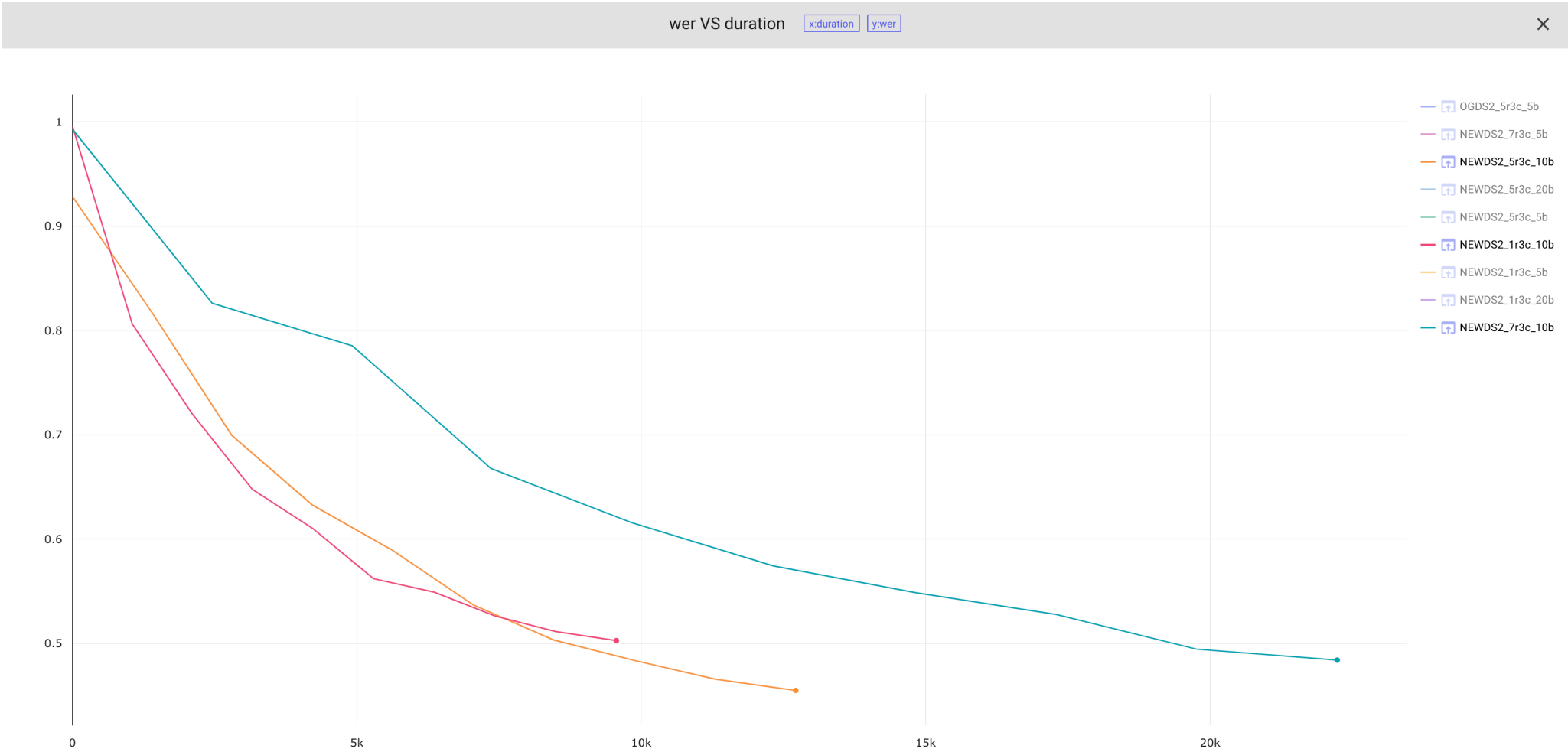
Results

- Smaller networks converge faster



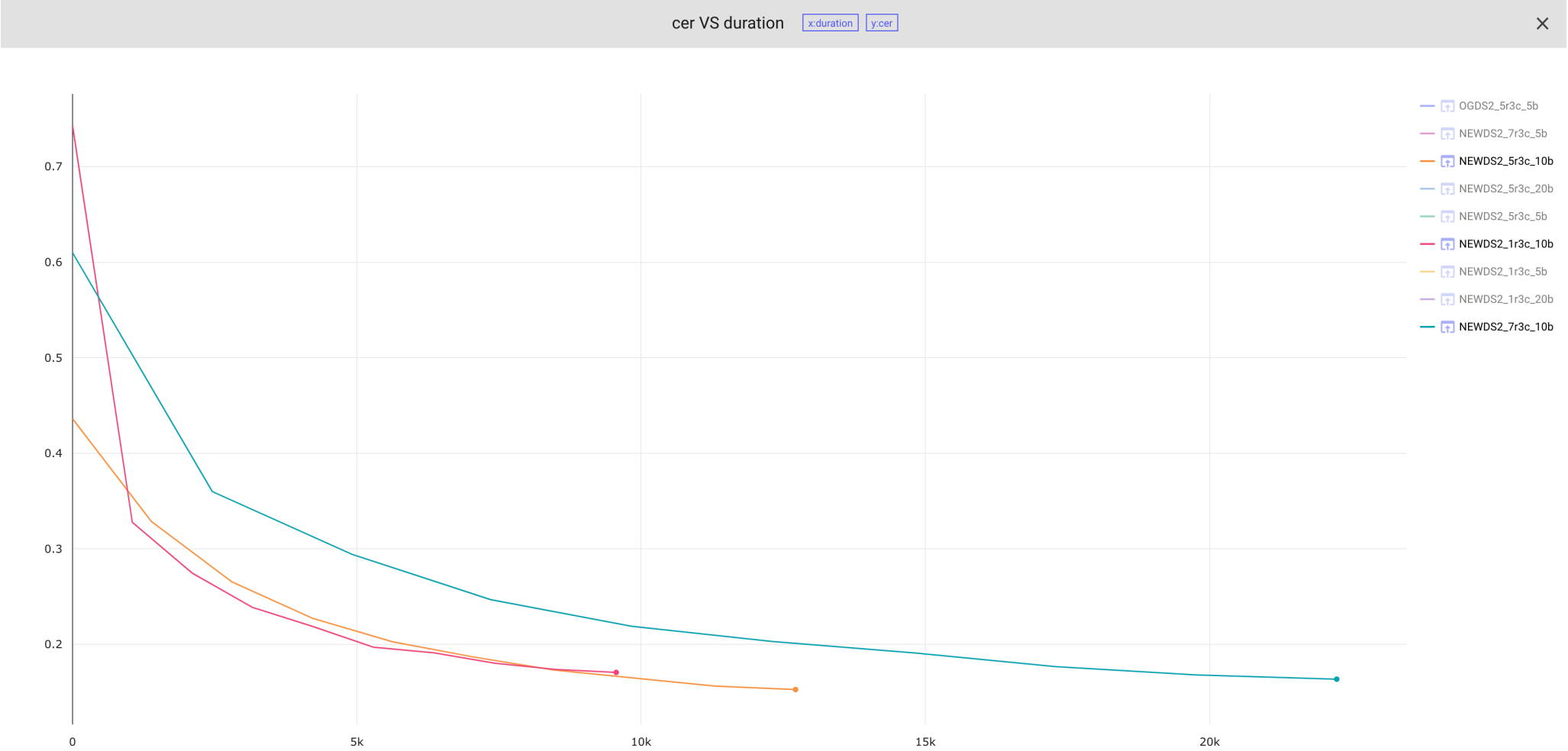
Results

- Network with 5 biGRU layers outperforms other networks in accuracy



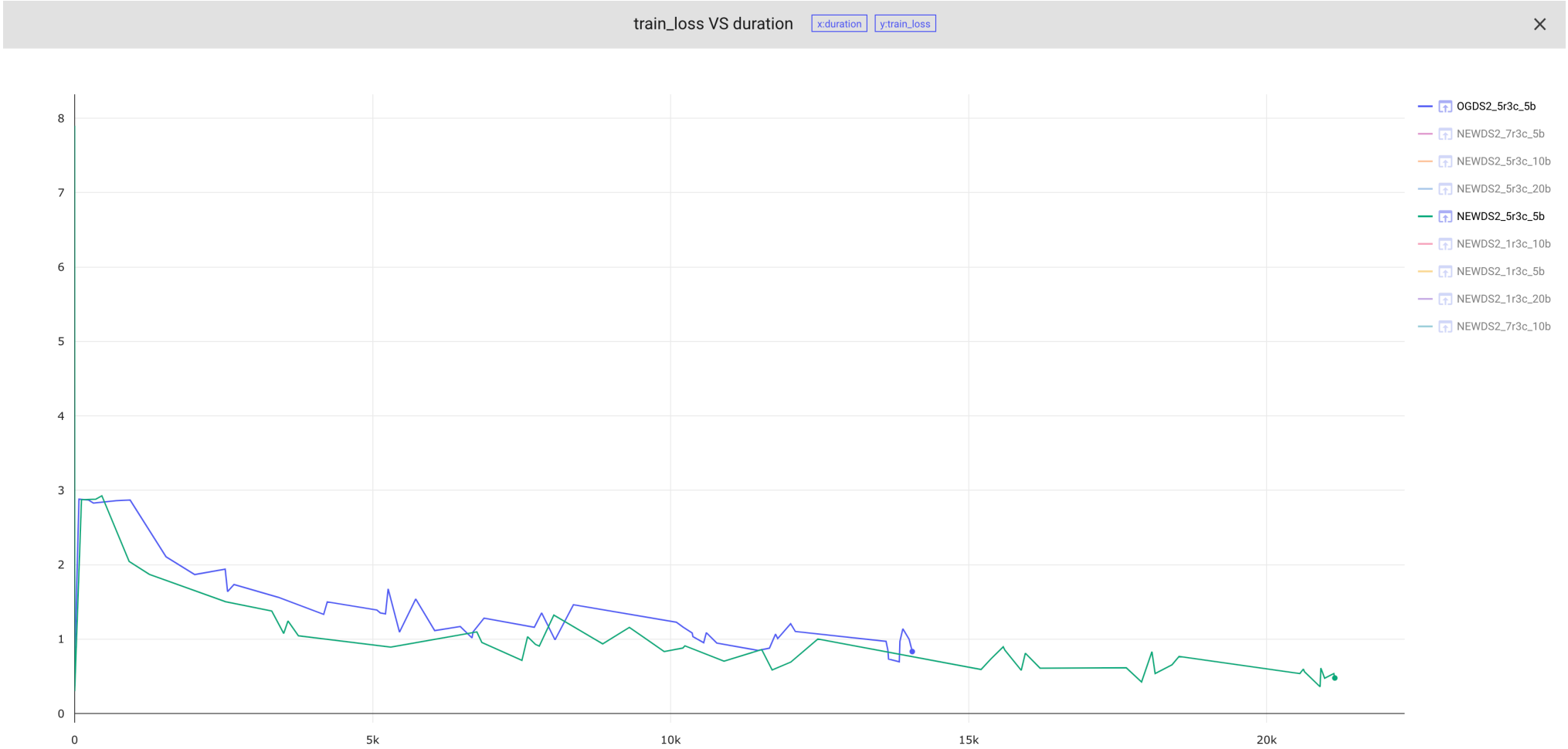
Results

- Network with 5 biGRU layers outperforms other networks in accuracy



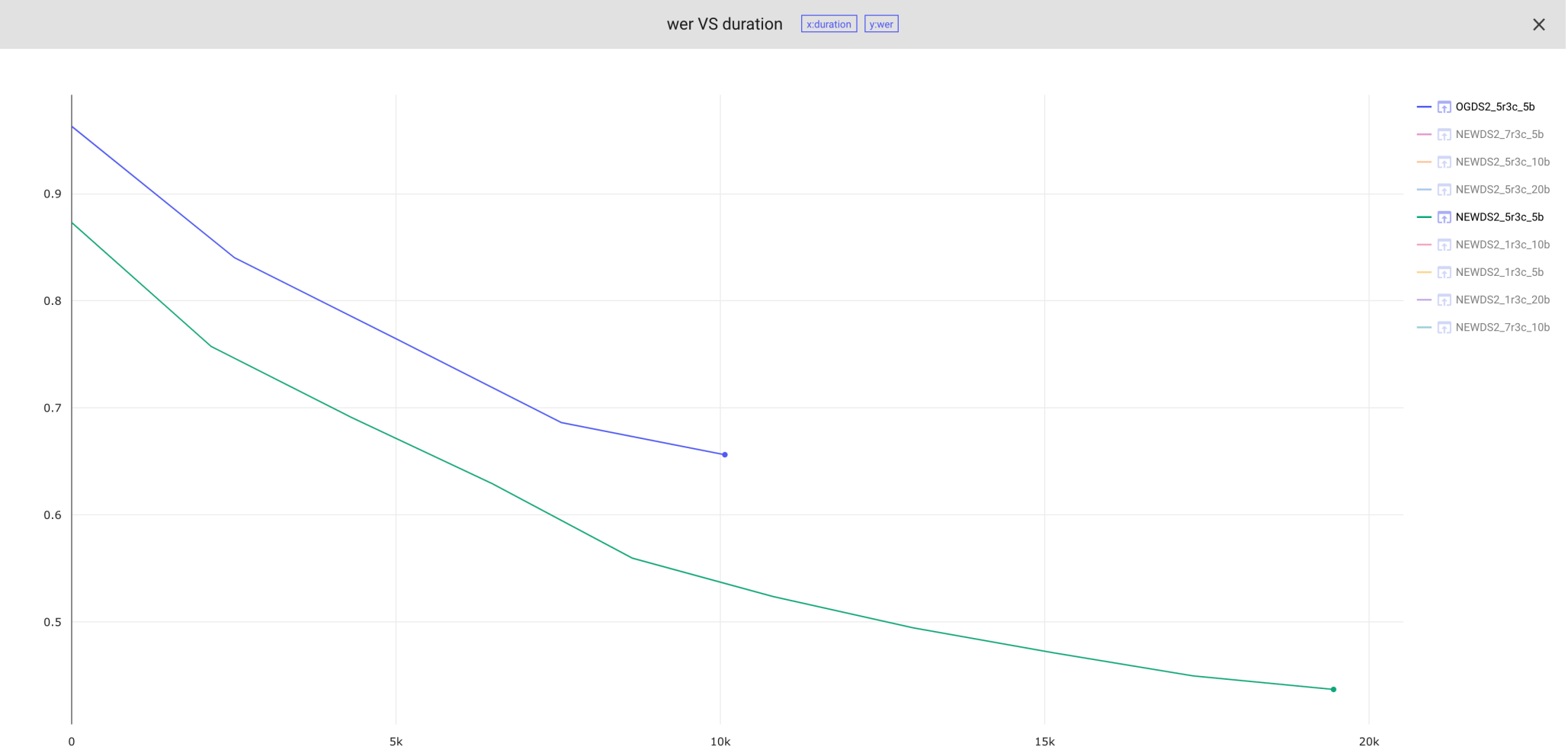
Results

- Residual Connections and Batch Normalization lead to faster convergence
  - Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



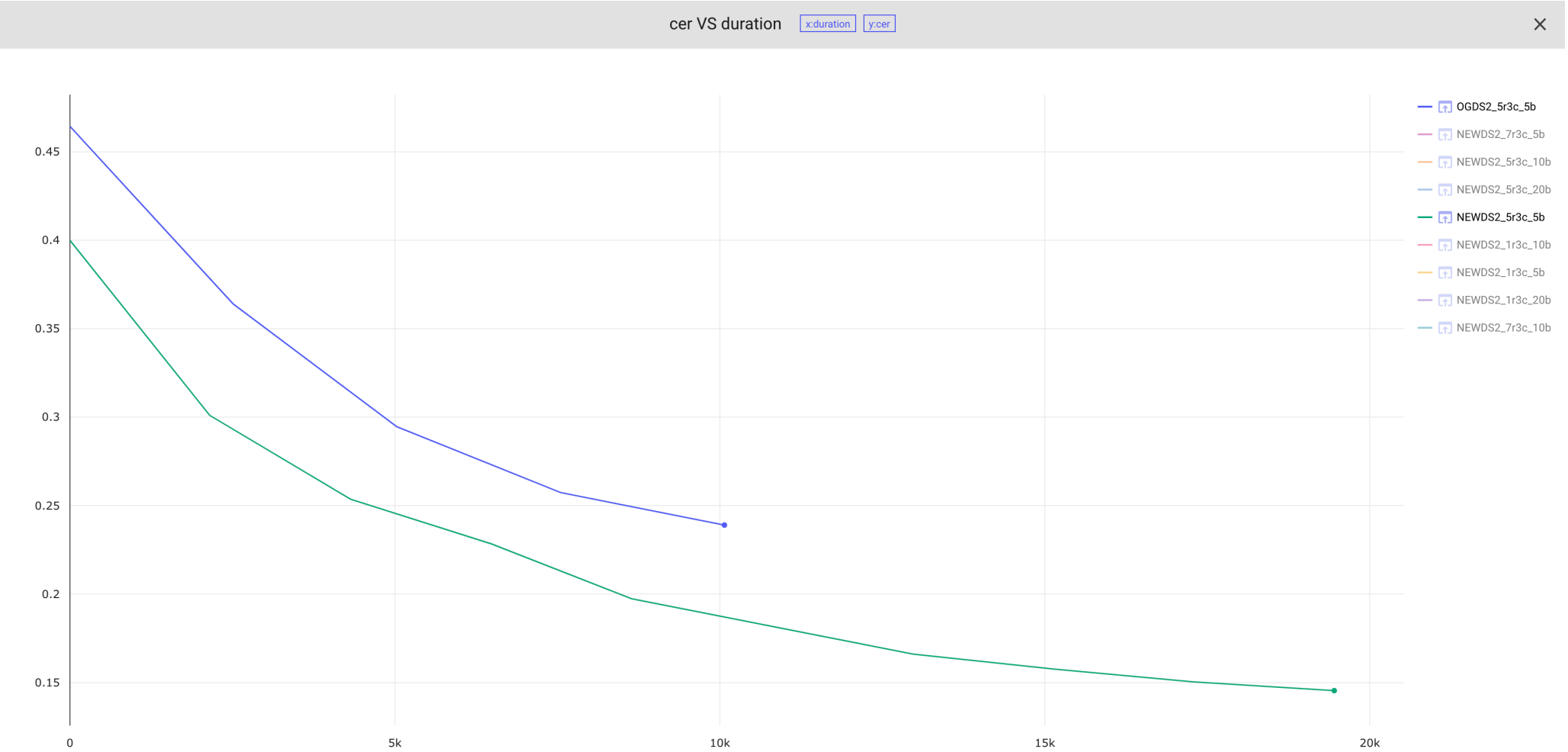
Results

- Residual Connections and Batch Normalization lead to better accuracy
  - Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



Results

- Residual Connections and Batch Normalization lead to better accuracy
  - Complete graph can be found here: https://www.comet.ml/yaygreat/deepspeech2/view/new



Next Steps

- More Testing:
- Apply to conversational and accented datasets
  - COMMONVOICE or VCTK datasets
- CER was always lower than WER
  - network architecture might be better suited for character-based languages such as Mandarin

- For better accuracy:
- Apply to larger datasets for longer training periods across multiple GPUs
- Use language model + beam search

- Look into future alternatives that are more computational efficient and that would not require an external language model

References

- Note: As stated above, this presentation is a work of fiction; the following are the actual inventors of the ideas described in this presentation

- D. Amodei, et. al., “Deep Speech 2: End-to-End Speech Recognition in English and Mandarin,” arXiv:1512.0259, 2015.

- A. Hannun, et. al., “Deep Speech: Scaling up end-to-end speech recognition,” arXiv:1412.5567, 2014.

- D. Hendrycks, et. al., “GAUSSIAN ERROR LINEAR UNITS (GELUS),” arXiv:1606.08415, 2018.

References

- Note: As stated above, this presentation is a work of fiction; the following are the actual inventors of the ideas described in this presentation

- H. Li, et. al., “Visualizing the Loss Landscape of Neural Nets,” arXiv:1712.09913, 2018.

- D. Park, et. al., “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition,” arXiv:1904.08779, 2019.

- S. Ioffe, et. al., “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” arXiv:1502.03167, 2015.

References

- Note: As stated above, this presentation is a work of fiction; the following are the actual inventors of the ideas described in this presentation

- T. Sterin, et. al., “An Intrinsic Difference Between Vanilla RNNs and GRU Models,” ISBN: 978-1-61208-531-9, 2017.

Disclaimer: This presentation is a work of fiction written from the perspective of a 2020 researcher traveling back in time to mid 2013 to share some 2020 xNN based application ideas; references to credit the actual inventors of the various ideas is provided at the end

Thank You!