TRANSFORMERS FOLLOW-UP: WHAT ABOUT AUDIO?

Jan Schlüter

43rd Vienna Deep Learning Meetup

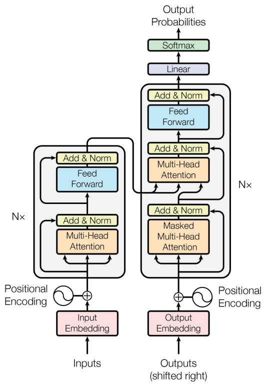
2021-12-01

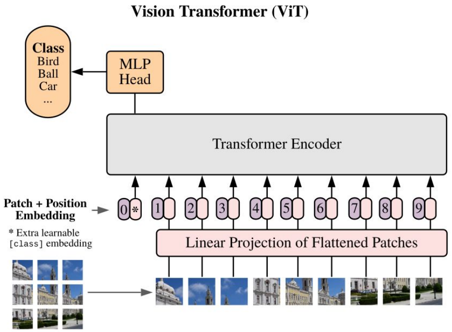
JOHANNES KEPLER UNIVERSITY LINZ

Institute of Computational Perception

STARTING POINT

- Transformers process a set of vectors (*tokens*) using *self-attention*
- Enriching tokens with position information allows processing structured data
- Originally proposed to process text as a sequence of words or word parts [1]
- Later used to process images as a grid of small image patches [2]

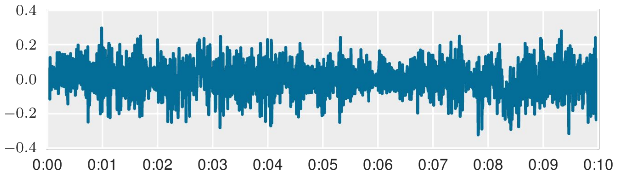


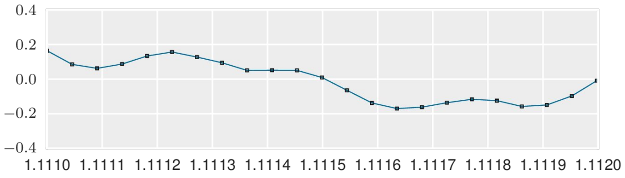


[1] Attention is all you need (arxiv.org/abs/1706.03762) [2] An image is worth 16x16 words (arxiv.org/abs/2010.11929)

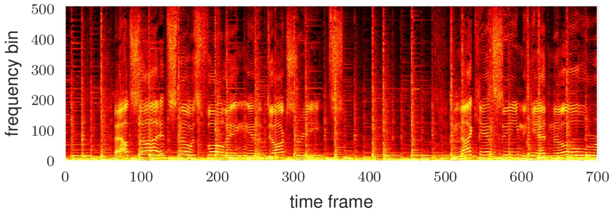
AUDIO DATA

- **Waveform:** Sound represented as pressure variations in a medium measured at a regular time interval (e.g., 44100 times per second, 2^16 possible values)





- **Spectrogram:** Sound represented as distribution of energy over frequencies at a regular time interval (e.g., 70 times per second, 513 frequency bands)



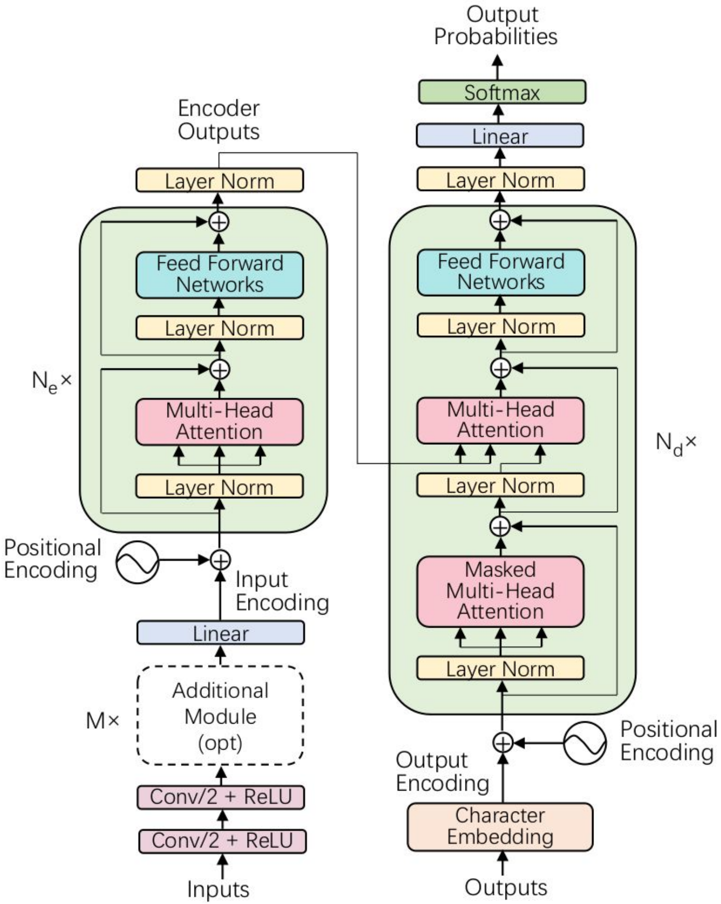
MODELING AUDIO DATA WITH TRANSFORMERS

- **Waveform:** Sound represented as pressure variations in a medium measured at a regular time interval (e.g., 44100 times per second, 2^16 possible values)
  - could use transformer with vocabulary of 2^16 tokens, but…
  - … the sequence length is *way* too large!

- **Spectrogram:** Sound represented as distribution of energy over frequencies at a regular time interval (e.g., 70 times per second, 513 frequencies, fp32)
  - much more manageable sequence length
  - we will look at different ideas to turn this into a set of tokens

SPEECH-TRANSFORMER

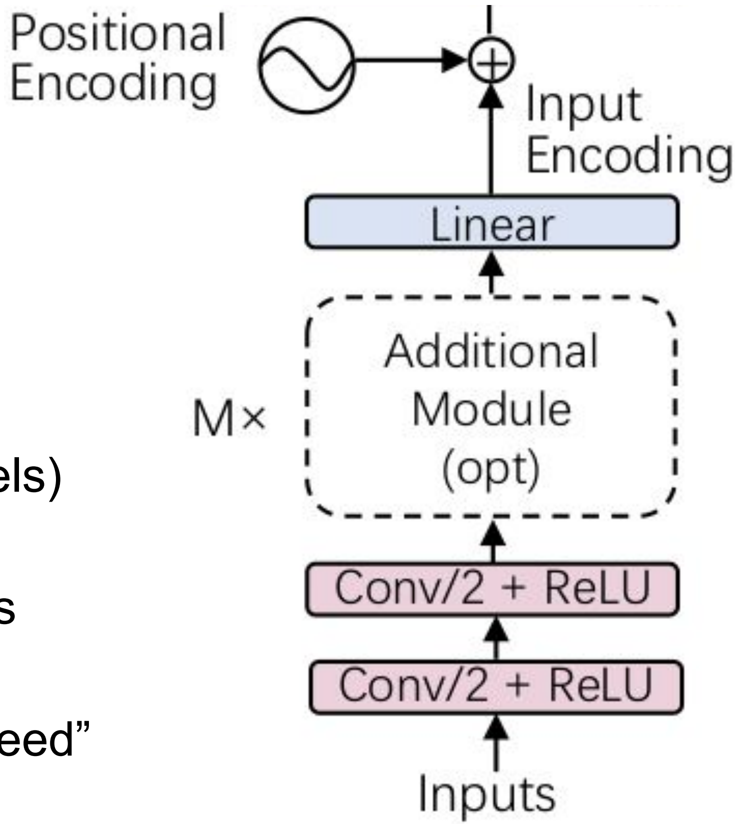
- Encoder-Decoder for Speech Recognition
- Input: spectrograms with 80 frequency bands, 100 frames per second + temporal derivatives



Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition (ICASSP 2018)

SPEECH-TRANSFORMER

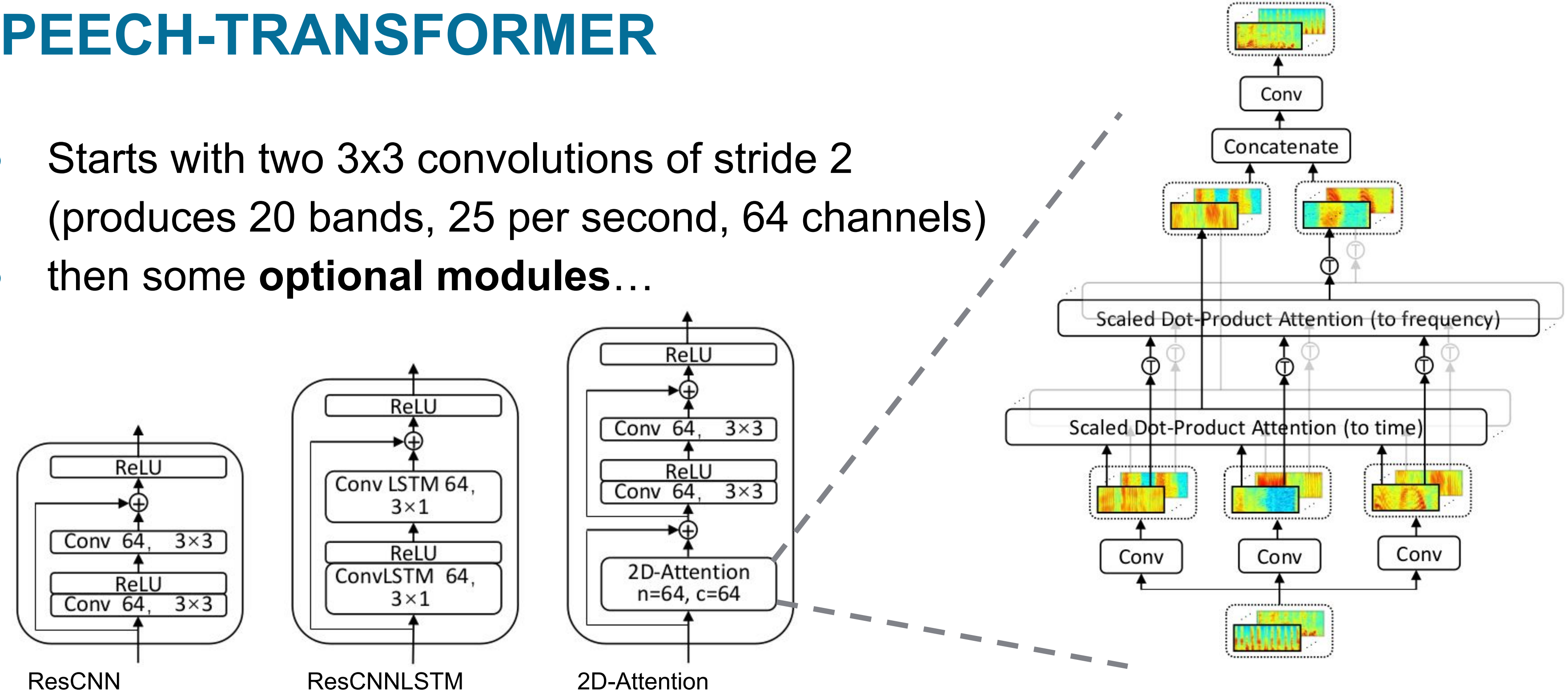
- Encoder-Decoder for Speech Recognition
- Input: spectrograms with 80 frequency bands, 100 frames per second + temporal derivatives
- Starts with two 3x3 convolutions of stride 2 (produces 20 bands, 25 per second, 64 channels)
- then some optional modules…
- and finally a linear projection to 256 dimensions (i.e., 25 tokens of 256 dimensions per second)
- Positional encoding as in “Attention is all you need”



Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition (ICASSP 2018)

SPEECH-TRANSFORMER

- Starts with two 3x3 convolutions of stride 2 (produces 20 bands, 25 per second, 64 channels)
- then some **optional modules**…



Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition (ICASSP 2018)

SPEECH-TRANSFORMER

- Evaluated on Wall Street Journal, eval92, Word Error Rate
- Note: Character-based, without language model
- Competitive with other such models of that time, trains faster than RNN

Model	WER
base model	12.20
base model + 4×ResCNN	11.90
base model + 4×ResCNNLSTM	12.01
base model + 2×1D-Attention	11.69
base model + 2×2D-Attention	**11.43**

2D-Attention helps!

Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition (ICASSP 2018)

SPEECH-TRANSFORMER

- Evaluated on Wall Street Journal, eval92, Word Error Rate
- Note: Character-based, without language model
- Competitive with other such models of that time, trains faster than RNN

Model	WER
base model	12.20
base model + 4×ResCNN	11.90
base model + 4×ResCNNLSTM	12.01
base model + 2×1D-Attention	11.69
base model + 2×2D-Attention	**11.43**
big model	**10.92**
big model + 2×2D-Attention	11.01

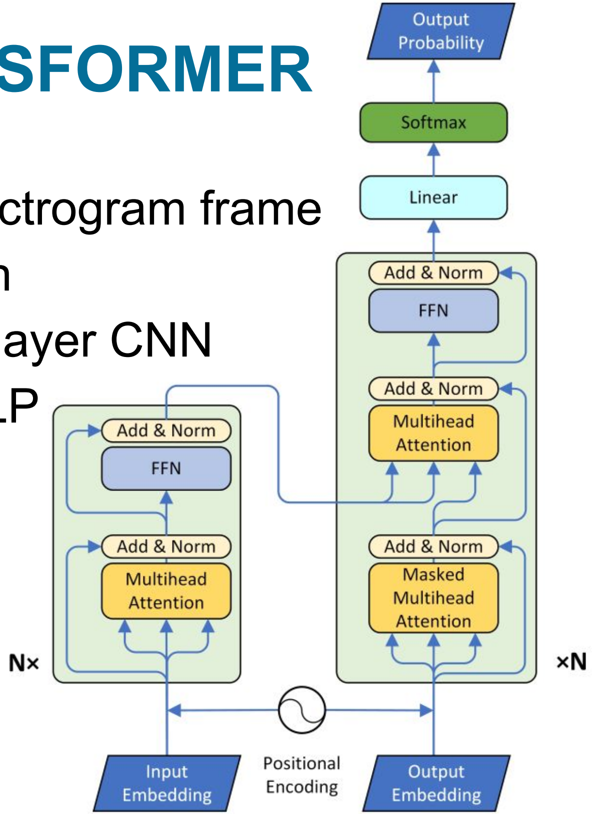
2D-Attention helps!

… not. (big model = larger inner dimension
in transformer feedforward part)

Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition (ICASSP 2018)

NEURAL SPEECH SYNTHESIS W/ TRANSFORMER

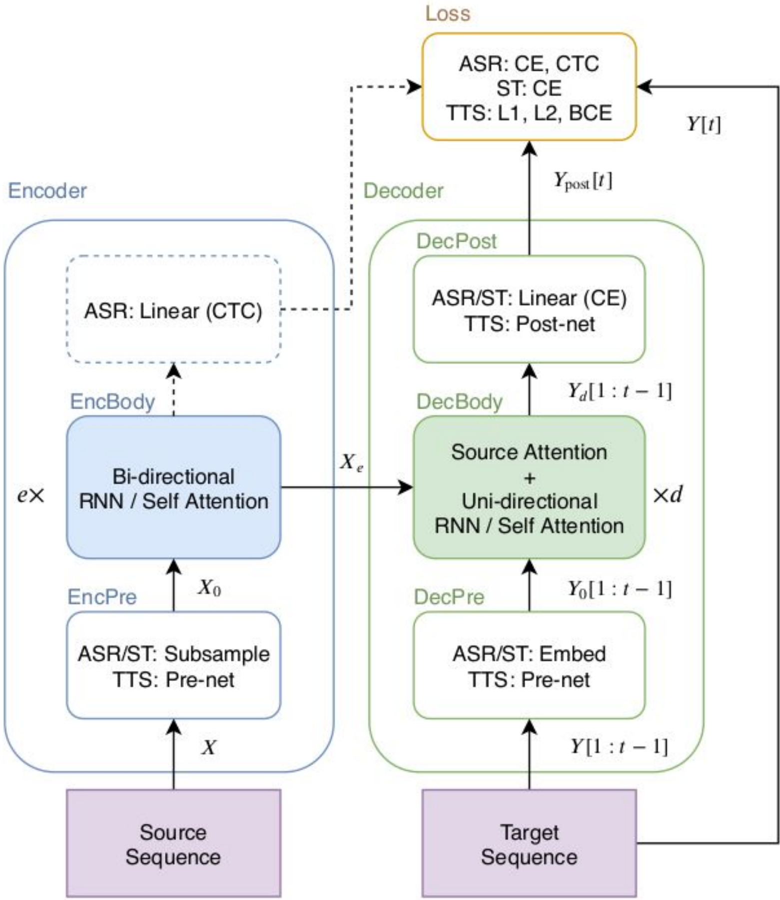
- Encoder reads phonemes, decoder predicts next spectrogram frame
- Inspired by Tacotron2, replacing LSTMs with attention
- 512-dim phoneme embeddings preprocessed with 3-layer CNN
- 80-dim spectrogram frames projected with 2-layer MLP
- Usual sin/cos positional encoding, added to embeddings with learnable weight to make up for possible differences in the two domains
- Results: Slightly better than Tacotron2, 4.25x faster to train due to better parallelization (no recurrence)



Neural Speech Synthesis with Transformer Network (AAAI 2019)

ESPNET

- Direct comparison of Transformer vs. LSTMs for Speech Recognition, Text-to-Speech, and Text Translation
- Transformer beats LSTMs for Speech Recognition, and performs similarly for Text-To-Speech
- Lessons on training transformers vs. RNNs: Transformers profit from larger minibatches and dropout, RNNs do not. Transformers are faster to train, but the decoder is slow!



A Comparative Study on Transformer vs RNN in Speech Applications (ASRU 2019)

JOHANNES KEPLER UNIVERSITY LINZ

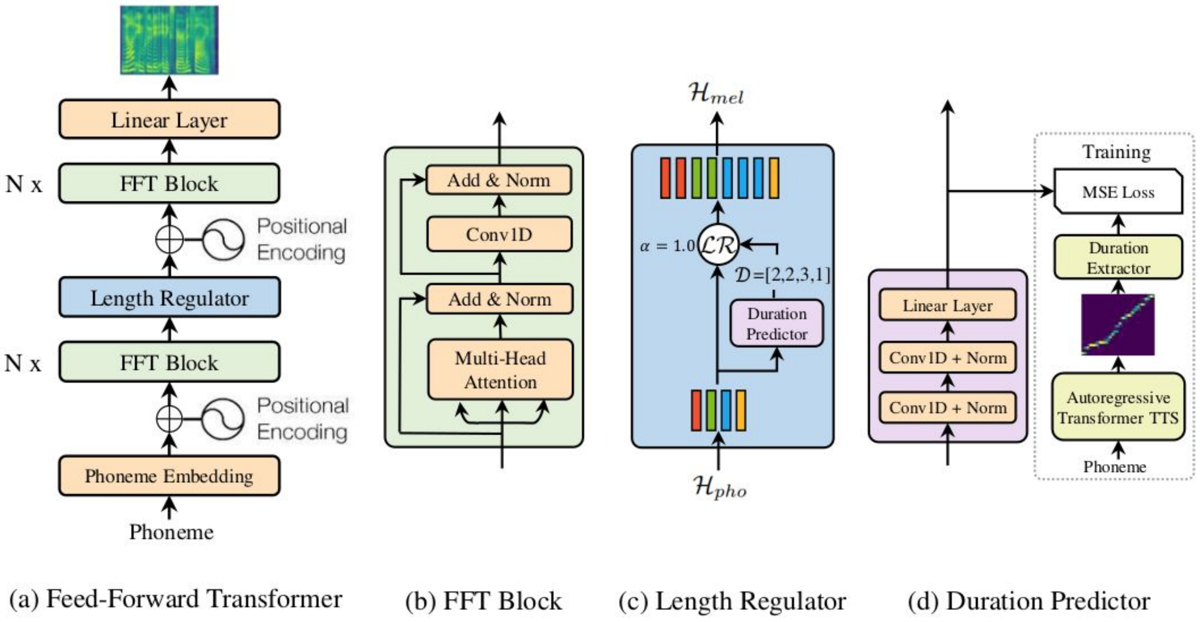
FASTSPEECH

- Having the decoder predict spectrogram frames autoregressively is slow
- Transformers can transform a sequence into another sequence
- Can we directly transform a character sequence to a speech spectrogram, fully parallelized, in one go?

FastSpeech: Fast, Robust and Controllable Text to Speech (NeurIPS 2019)

FASTSPEECH

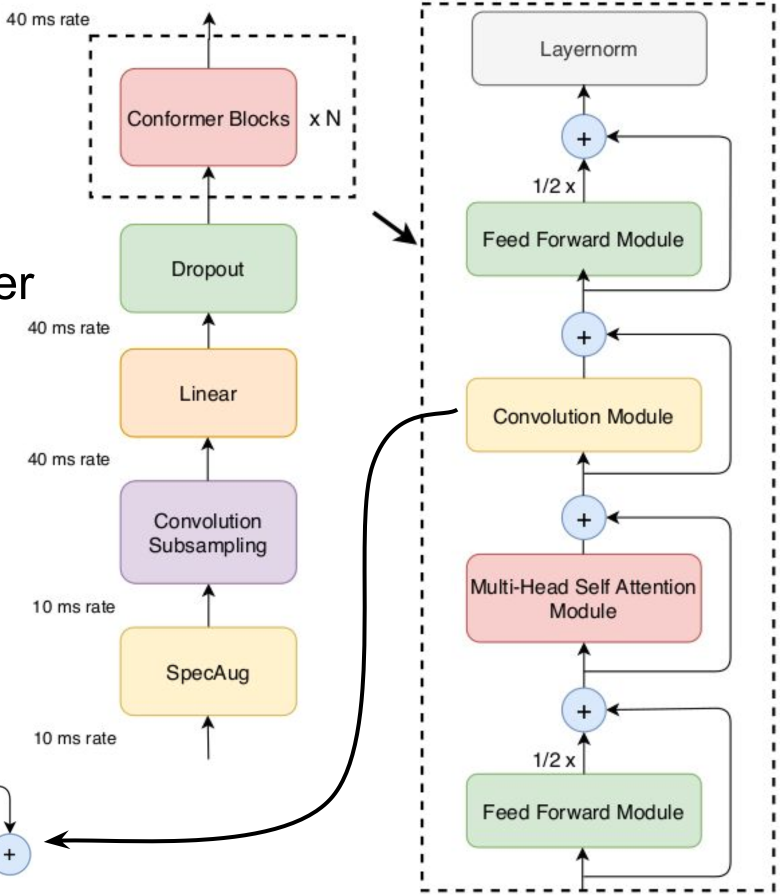
- Main trick: Repeat input phonemes to match length of output sequence
- Requires a phoneme duration predictor, which is trained to match attention spans of an autoregressive Transformer TTS

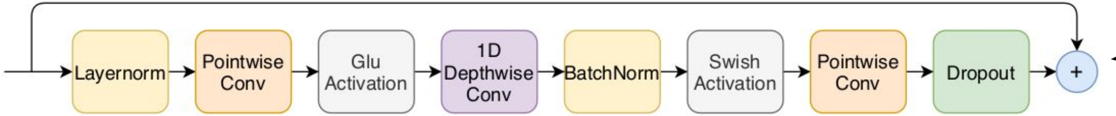


FastSpeech: Fast, Robust and Controllable Text to Speech ([NeurIPS 2019](#))

CONFORMER

- Speech Recognition with modified Transformer
- Input: usual 80-dim, 100 frames/sec spectrogram, two 3x3, stride 2 convolutions, linear projection to 256-dimensional frames
- Applies 1D convolutions after self-attention
- Other tricks: “Macaron-style” splitting of Feed Forward Module in two parts, relative positional encoding from Transformer-XL

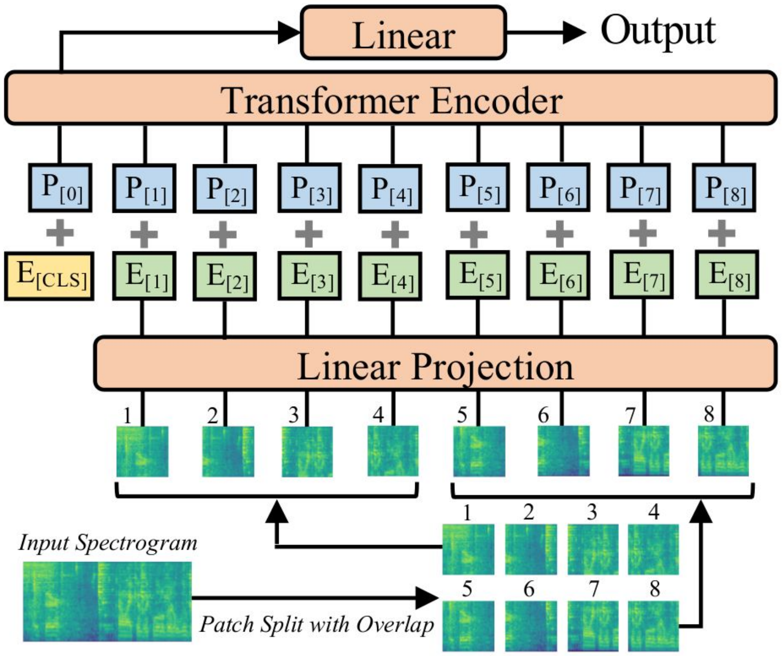




Conformer: Convolution-augmented Transformer for Speech Recognition (Interspeech 2020)

AUDIO SPECTROGRAM TRANSFORMER

- 632-way classification of audio clips (AudioSet, 2 million YouTube excerpts)
- Instead of treating spectrogram as temporal 1D sequence, treat as 2D image
- Extract 16x16 patches, apply linear projection to 1024 dims, add 2D trainable positional embedding
- Prepend classification token “CLS”, pass through transformer, predict class from transformed CLS token

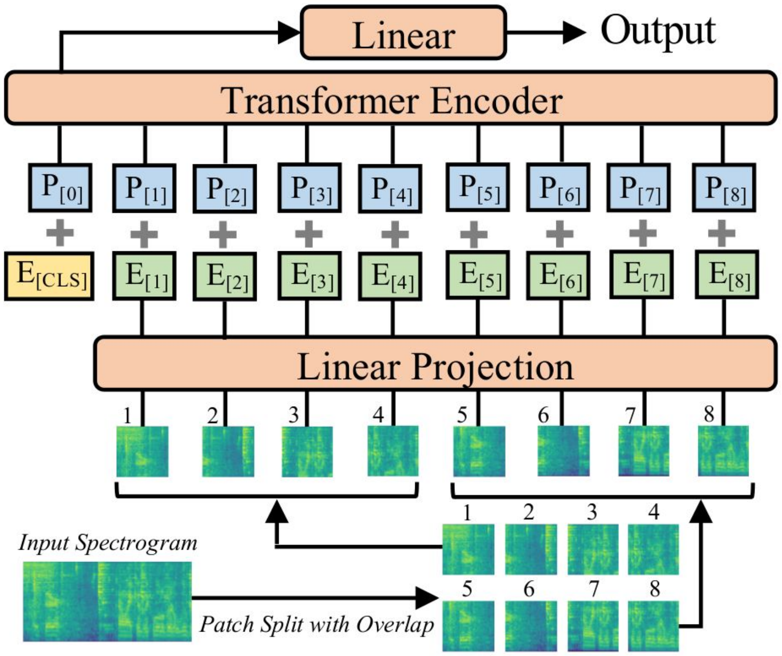


AST: Audio Spectrogram Transformer ([Interspeech 2021](#))

JOHANNES KEPLER UNIVERSITY LINZ

AUDIO SPECTROGRAM TRANSFORMER

- Extract 16x16 patches, apply linear projection to 768 dims, add 2D trainable positional embedding
- Same as Vision Transformer (ViT)
- Can use weights of ViT from ImageNet!
  - Vision transformer: 384x384 input pixels, results in 24x24 patches of size 16x16
  - AST: 128x1000 input pixels for 10 sec., results in 12x100 patches of size 16x16 extracted at stride 10x10
  - Interpolate positional embeddings from 24x24 to 12x100
  - profit



AST: Audio Spectrogram Transformer (Interspeech 2021)

JOHANNES KEPLER UNIVERSITY LINZ

AUDIO SPECTROGRAM TRANSFORMER

- ImageNet pretraining helps:

	Balanced Set	Full Set
No Pretrain	0.148	0.366
ImageNet Pretrain (Used)	0.347	0.459

- Using the embedding helps:

	Balanced Set
Reinitialize	0.305
Nearest Neighbor Interpolation	0.346
Bilinear Interpolation (Used)	0.347

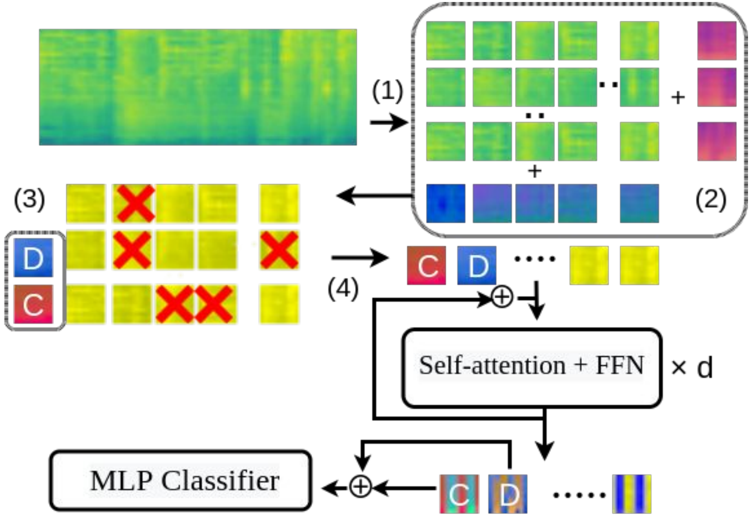
- DeiT outperforms ViT:

	# Params	ImageNet	AudioSet
ViT Base [11]	86M	0.846	0.320
ViT Large [11]*	307M	0.851	0.330
DeiT w/o Distill [12]	86M	0.829	0.330
DeiT w/ Distill (Used)	87M	0.852	0.347

AST: Audio Spectrogram Transformer (Interspeech 2021)

PATCHOUT FAST SPECTROGRAM TRANSFORMER

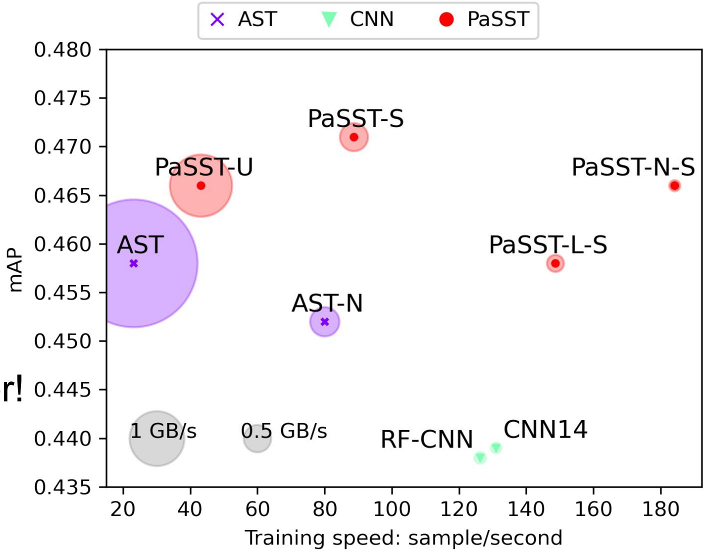
- Due to the large number of patches, AST requires 4 GPUs à 12 GiB to train
- Can we train the model on a single consumer GPU?
- Remember: Transformers do not actually process sequences of tokens, but sets of tokens with positional embeddings
- PatchOut: Omit a random subset of tokens during training



Efficient Training of Audio Transformers with Patchout (arxiv.org/abs/2110.05069)

PATCHOUT FAST SPECTROGRAM TRANSFORMER

- Some different variants:
  - U = unstructured: completely at random
  - S = structured: time and frequency slices
  - N = no overlap of patches (even fewer patches)
  - L = lightweight (remove every other Transformer block)
- Outperforms both CNNs and AST
  - PatchOut is not only faster, but functions as a regularizer!
- Fits on a single RTX 2080 Ti
- New state of the art on AudioSet



Efficient Training of Audio Transformers with Patchout (arxiv.org/abs/2110.05069)

JOHANNES KEPLER UNIVERSITY LINZ

PATCHOUT FAST SPECTROGRAM TRANSFORMER

- Some different variants:
  - U = unstructured: completely at random
  - S = structured: time and frequency slices
  - N = no overlap of patches
  - L = lightweight
- Outperforms both CNNs and AST
  - PatchOut is not only faster, but better!
- Fits on a single RTX 2080 Ti
- New state of the art on AudioSet
- Note: AST and PASST pretrained on AudioSet can also be used for downstream tasks

	OpenMIC	ESC50	DCASE20
Baseline	.795 [19]	76.9 [20]	54.1 [21]
State-of-the-Art	.831 [16]	95.6 [5]	73.7 [22]
PaSST-B ⋆	.837	96.3	**76.3**
PaSST-U ⋆	**.843**	96.5	75.6
PaSST-S ⋆	**.843**	**96.8**	75.6
PaSST-S-L ⋆	.841	95.5	73.7
PaSST-S-N ⋆	.840	96.4	73.9

Efficient Training of Audio Transformers with Patchout (arxiv.org/abs/2110.05069)

JOHANNES KEPLER UNIVERSITY LINZ

MODELING AUDIO DATA WITH TRANSFORMERS

- **Waveform:** Sound represented as pressure variations in a medium measured at a regular time interval (e.g., 44100 times per second, 2^16 possible values)
  - could use transformer with vocabulary of 2^16 tokens, but…
  - … the sequence length is *way* too large!

MODELING AUDIO DATA WITH TRANSFORMERS

- **Waveform:** Sound represented as pressure variations in a medium measured at a regular time interval (e.g., 44100 times per second, 2^16 possible values)
  - could use transformer with vocabulary of 2^16 tokens, but…
  - … the sequence length is *way* too large!
  - can we compress it though?

JUKEBOX

- **Waveform:** Sound represented as pressure variations in a medium measured at a regular time interval (e.g., 44100 times per second, 2^16 possible values)
- Apply 2M parameters VQ-VAE to learn discrete codes, resulting sequence has 345 tokens per second, 2048 possible values
- Apply 5B parameters Transformer to learn language model
  - Conditioned on artist, genre and lyrics
- Decide on conditioning ⇒ sample from Transformer ⇒ decode with VQ-VAE
  - See openai.com/blog/jukebox/ for audio examples
- Transformer representation also useful for downstream classification tasks! (ISMIR 2021)

Jukebox: A Generative Model for Music (OpenAI)