Multitask Learning with Low-Level Auxiliary Tasks

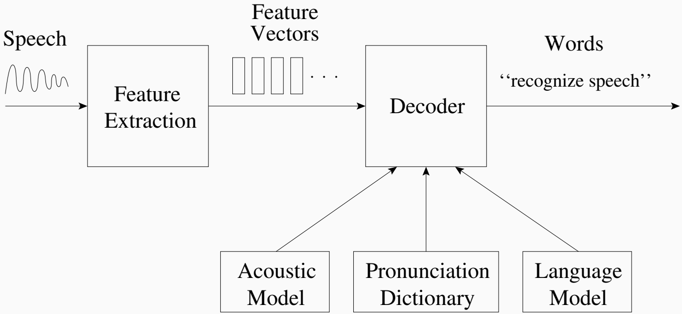
for Speech Recognition

Shubham Toshniwal, Hao Tang, Liang Lu, Karen Livescu

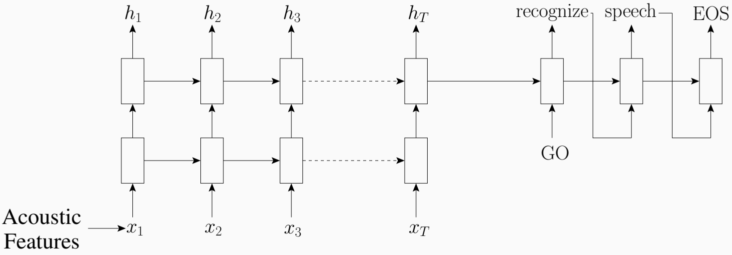
Toyota Technological Institute at Chicago

Conventional ASR Systems

- Traditional automatic speech recognition (ASR) systems are modular.
- Different components of the system are trained separately.
- Components correspond to different levels of representation - frame-level states, phones, and words etc.



- Neural end-to-end models for ASR have become viable and popular.
- End-to-end models are appealing because:
    - Conceptually simple; all model parameters contribute to the same final goal.
    - Impressive results in ASR (Zweig et al. 2016) as well as other domains (Vinyals et al. 2015, Huang et al. 2016).

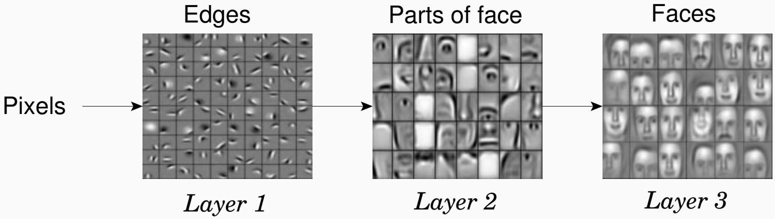


End-to-end Models: Cons

However, end-to-end models have some drawbacks as well:

- Optimization can be challenging.
- Ignore potentially useful domain-specific information about intermediate representations, as well as existing intermediate levels of supervision.
- Hard to interpret intermediate learned representations, thus harder to debug.

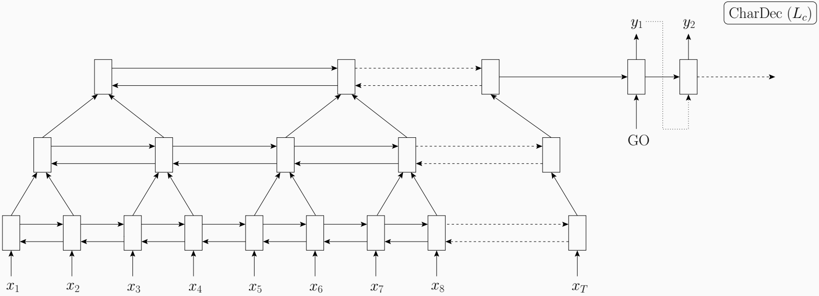
- Analysis of some deep end-to-end models found that different layers tend to specialize for different sub-tasks (Mohamed et al. 2012, Zeiler et al. 2014).
- Lower layers focus on lower-level representation and higher ones on higher-level representation.



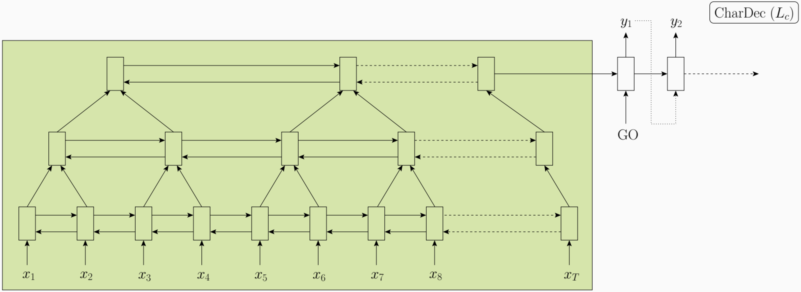
Motivation

- Can we encourage such intermediate representation learning more explicitly ?
- Multitask learning: Combine final task loss (speech recognition) with losses corresponding to lower-level tasks (such as phonetic recognition) applied on lower layers (Søgaard et al. 2016).

Encoder-Decoder Model for speech recognition

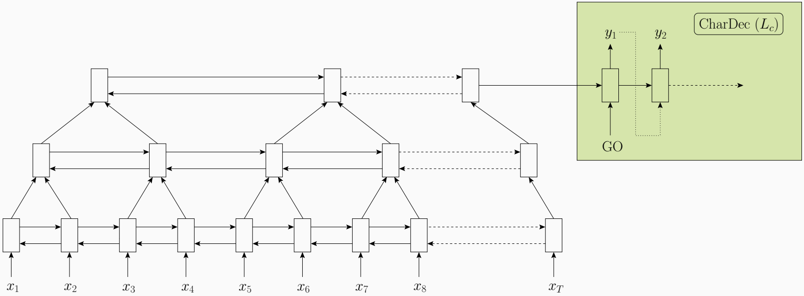


- We use the attention-enabled encoder-decoder variant proposed by Chan et al. 2015.



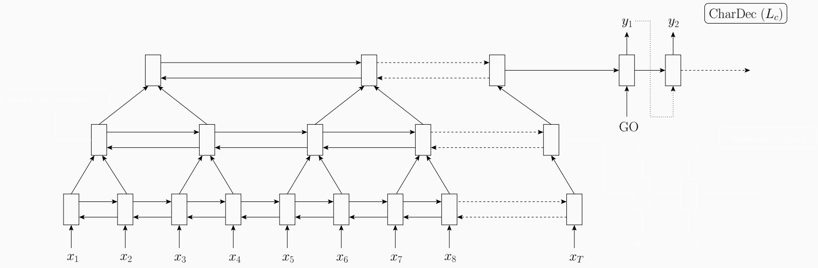
- We use the attention-enabled encoder-decoder variant proposed by Chan et al. 2015.
- *Speech encoder*: A pyramidal bidirectional LSTM that:
  (i) Reads in acoustic features $\mathbf{x} = (x_1, \ldots, x_T)$
  (ii) Outputs a sequence of high-level features (hidden states).

Encoder-Decoder Model for speech recognition



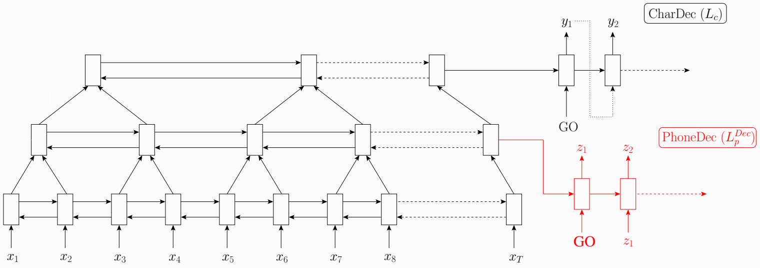
- We use the attention-enabled encoder-decoder variant proposed by Chan et al. 2015.
- *Speech encoder*: A pyramidal bidirectional LSTM that:
  (i) Reads in acoustic features $\boldsymbol{x} = (x_1, \ldots, x_T)$
  (ii) Outputs a sequence of high-level features (hidden states).
- *Character decoder*: Attends to high-level features generated by *encoder* and outputs $\boldsymbol{y} = (y_1, \ldots, y_K)$.

Adding Phoneme Supervision



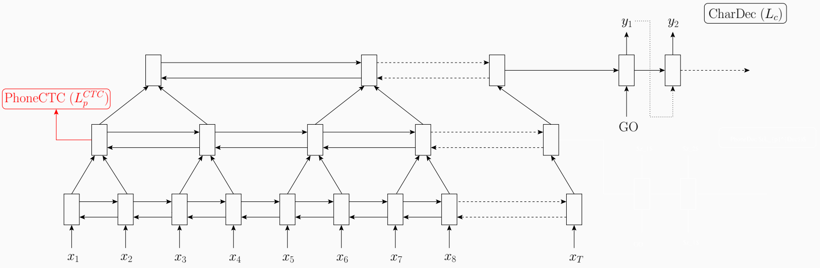
- Phoneme-level supervision obtained using pronunciation dictionary.

Adding Phoneme Supervision



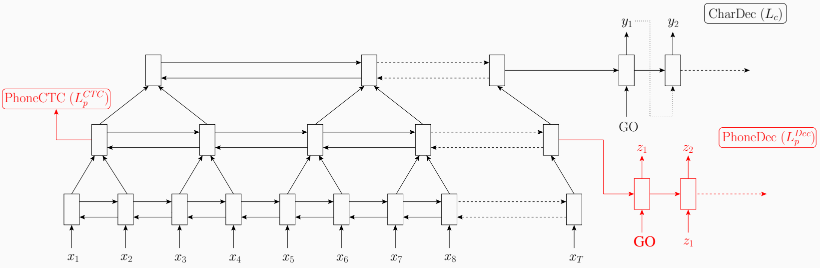
- Phoneme-level supervision obtained using pronunciation dictionary.
- Experiment with two types of sequence loss:
  (a) Phoneme Decoder Loss (L_p^{Dec}),

Adding Phoneme Supervision

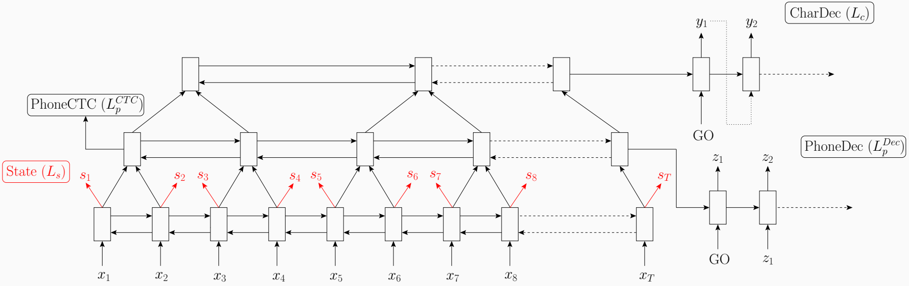


- Phoneme-level supervision obtained using pronunciation dictionary.
- Experiment with two types of sequence loss:
  (a) Phoneme Decoder Loss (L_p^{Dec}),
  (b) CTC-loss (L_p^{CTC})

Adding Phoneme Supervision



- Phoneme-level supervision obtained using pronunciation dictionary.
- Experiment with two types of sequence loss:
  (a) Phoneme Decoder Loss (L_p^{Dec}),
  (b) CTC-loss (L_p^{CTC})
- Training Loss L is given by: $L = \frac{1}{2}(L_c + L_p)$.



- We also experiment with frame-level state supervision.
- Training Loss L is then: $L = \frac{1}{3}(L_c + L_p + L_s)$.

Dataset & Model Details

Dataset:

- Switchboard corpus - 300 hrs of conversational speech data.
- Standard training/development/test split is used.

Model:

- *Speech Encoder*: 4-layer pyramidal bidirectional LSTM.
- *Character Decoder*: 1-layer unidirectional LSTM.
- Both have 256 hidden units.

Dev Results

Table 1: Character error rate (CER) and word error rate (WER) results on development data.

Model	Dev CER (in %)	Dev WER (in %)
Enc-Dec (baseline)	14.6	26.0

Dev Results

Table 1: Character error rate (CER) and word error rate (WER) results on development data.

Model	Dev CER (in %)	Dev WER (in %)
Enc-Dec (baseline)	14.6	26.0
Enc-Dec + PhoneCTC-3	14.0	25.3
Enc-Dec + PhoneDec-3	13.8	24.9

Dev Results

Table 1: Character error rate (CER) and word error rate (WER) results on development data.

Model	Dev CER (in %)	Dev WER (in %)
Enc-Dec (baseline)	14.6	26.0
Enc-Dec + PhoneCTC-3	14.0	25.3
Enc-Dec + PhoneDec-3	13.8	24.9
Enc-Dec + PhoneDec-4	14.5	25.9

Table 1: Character error rate (CER) and word error rate (WER) results on development data.

Model	Dev CER (in %)	Dev WER (in %)
Enc-Dec (baseline)	14.6	26.0
Enc-Dec + PhoneCTC-3	14.0	25.3
Enc-Dec + PhoneDec-3	13.8	24.9
Enc-Dec + PhoneDec-4	14.5	25.9
Enc-Dec + State-2	13.6	24.1

Dev Results

Table 1: Character error rate (CER) and word error rate (WER) results on development data.

Model	Dev CER (in %)	Dev WER (in %)
Enc-Dec (baseline)	14.6	26.0
Enc-Dec + PhoneCTC-3	14.0	25.3
Enc-Dec + PhoneDec-3	13.8	24.9
Enc-Dec + PhoneDec-4	14.5	25.9
Enc-Dec + State-2	13.6	24.1
Enc-Dec + PhoneDec-3 + State-2	13.4	24.1

Table 2: WER (%) on test data for different end-to-end models.

Model	SWB	CHE	Full
Our models			
Enc-Dec (baseline)	25.0	42.4	33.7
Enc-Dec + PhoneDec-3 + State-2	23.1	40.8	32.0

Test Results

Table 2: WER (%) on test data for different end-to-end models.

Model	SWB	CHE	Full
Our models			
Enc-Dec (baseline)	25.0	42.4	33.7
Enc-Dec + PhoneDec-3 + State-2	23.1	40.8	32.0
Lu et al. 2016			
Enc-Dec	27.3	48.2	37.8
Enc-Dec (word) + 3-gram	25.8	46.0	36.0

Test Results

Table 2: WER (%) on test data for different end-to-end models.

Model	SWB	CHE	Full
Our models			
Enc-Dec (baseline)	25.0	42.4	33.7
Enc-Dec + PhoneDec-3 + State-2	23.1	40.8	32.0
Lu et al. 2016			
Enc-Dec	27.3	48.2	37.8
Enc-Dec (word) + 3-gram	25.8	46.0	36.0
Maas et al. 2015			
CTC	38.0	56.1	47.1

Table 2: WER (%) on test data for different end-to-end models.

Model	SWB	CHE	Full
Our models			
Enc-Dec (baseline)	25.0	42.4	33.7
Enc-Dec + PhoneDec-3 + State-2	23.1	40.8	32.0
Lu et al. 2016			
Enc-Dec	27.3	48.2	37.8
Enc-Dec (word) + 3-gram	25.8	46.0	36.0
Maas et al. 2015			
CTC	38.0	56.1	47.1
Zweig et al. 2016			
Iterated CTC	24.7	37.1	—

How does Multitask Learning help ?

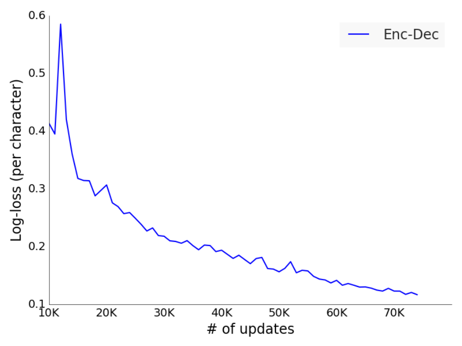


Figure 1: Log-loss of training data (only L_c) for different model variations.

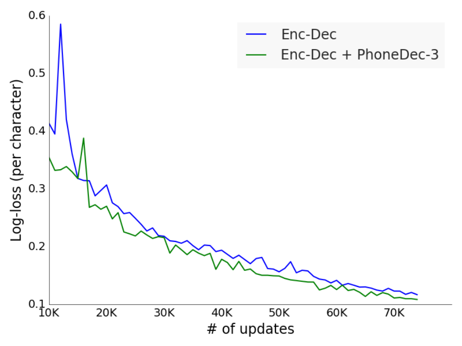


Figure 1: Log-loss of training data (only L_c) for different model variations.

How does Multitask Learning help ?

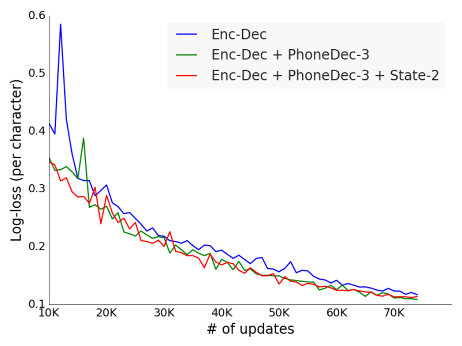


Figure 1: Log-loss of training data (only L_c) for different model variations.

Conclusion & Future Work

- Multitask learning is great!

Conclusion & Future Work

- Multitask learning is great!
- Using lower level supervision at lower-levels is the key to our gains.

- Multitask learning is great!
- Using lower level supervision at lower-levels is the key to our gains.

- More generally, our ASR model can be extended to incorporate higher-level supervision, such as semantic/syntactic labels.

Conclusion & Future Work

- Multitask learning is great!
- Using lower level supervision at lower-levels is the key to our gains.

- More generally, our ASR model can be extended to incorporate higher-level supervision, such as semantic/syntactic labels.
- The idea of incorporating different types of supervision at different levels is of broad interest (Hashimoto et al. 2016, Weiss et al. 2017, Rao et al. 2017).

Questions?