Deep Learning for Automatic Speech Recognition – Part III

Xiaodong Cui

IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

Fall, 2018

Outline

- End-to-end acoustic modeling
  - Connectionist Temporal Classification (CTC)
  - Encoder-decoder attention models
- Other techniques in acoustic modeling
  - Data augmentation
  - Speaker adaptation (transfer learning)
  - Multilingual acoustic modeling

Two Streams of DNN Acoustic Models

- Hybrid DNNs (discussed last lecture)
  - Commonly referred to as DNN-HMM or CD-DNN-HMM
  - Use GMM-HMM alignments as labels
  - Use dictionary and language model for decoding.
- End-to-end (E2E) DNNs
  - Directly deal with sequence-to-sequence mapping problem with unequal sequence lengths
  - Do not need alignments, dictionary and language model **in principle**.
  - Two E2E architectures:
    - Connectionist Temporal Classification (CTC)
    - Encoder-Decoder Attention models

Connectionist Temporal Classification (CTC)

Mathematical Formulation:

- Input: Observation sequence $X = \{x_1, x_2, \cdots, x_\mathsf{T}\}$
- Label: Target sequence $Z = \{z_1, z_2, \cdots, z_\mathsf{M}\}$
- Unequal lengths: $M < T$
- Model: A neural network with a softmax output layer

$$Z = \mathcal{N}_\lambda(X)$$

- Loss function: Maximum likelihood

$$\lambda^* = \underset{\lambda}{\operatorname{argmax}} \log P_\lambda(Z|X)$$

CTC Paths

- Allowing blanks and repeated labels ($\boldsymbol{L}' = \boldsymbol{L} \cup \{\square\}$)

$$\boldsymbol{\mathcal{B}}(a\square aabb\square\square) = \boldsymbol{\mathcal{B}}(\square aa\square\square abb) = ab$$

- Same length as the input sequence
- Many-to-one mapping
- Likelihood of the path π (conditional independency)

$$P(\pi|X) = \prod_{t=1}^{T} y^t_{\pi_t}, \quad \forall \pi \in \boldsymbol{L}'^{\mathsf{T}}$$

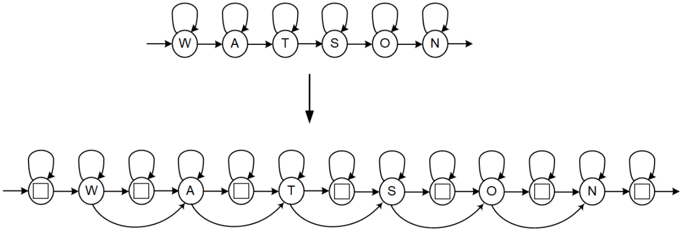
where $\pi_t = k$ and y^t_k is the output of the softmax layer, output unit k at time t.

- Likelihood of the label sequence

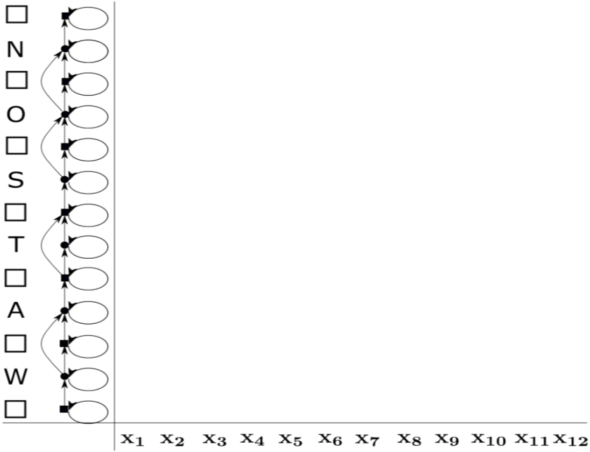
$$P(Z|X) = \sum_{\pi \in \boldsymbol{\mathcal{B}}^{-1}(Z)} p(\pi|X)$$

The CTC Forward-Backward Algorithm

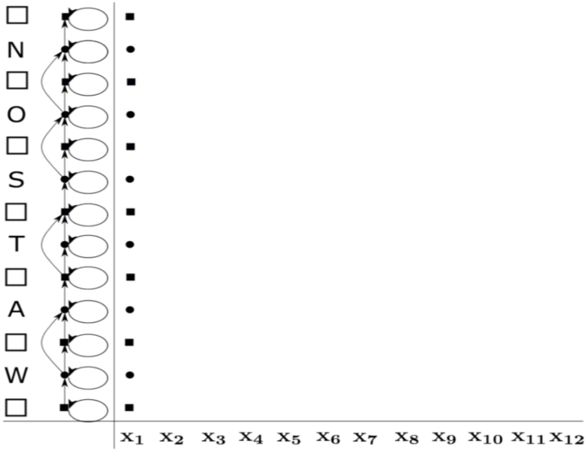
- Define a modified label sequence Z'
  - add blanks to the beginning and the end of the original label sequence Z
  - insert blanks between every pair of labels
  - $|Z'| = 2|Z| + 1$



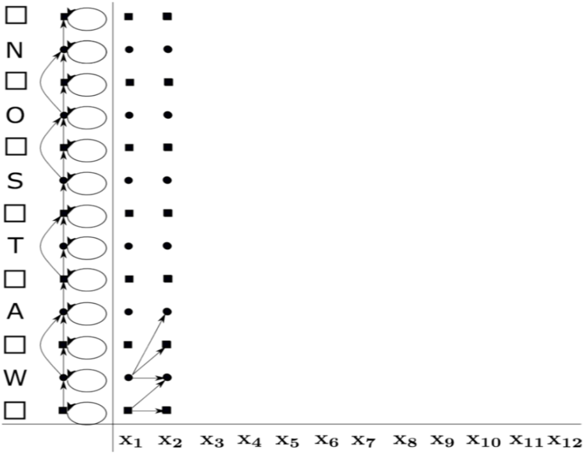
The CTC Forward-Backward Algorithm



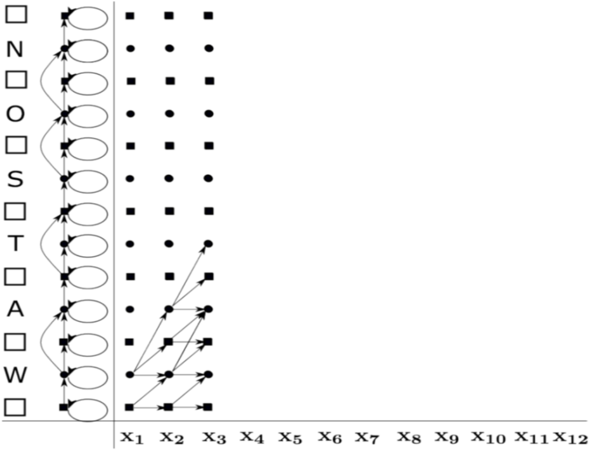
The CTC Forward-Backward Algorithm



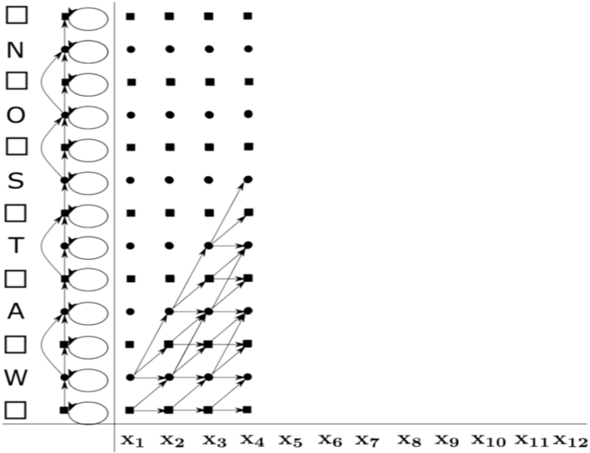
The CTC Forward-Backward Algorithm



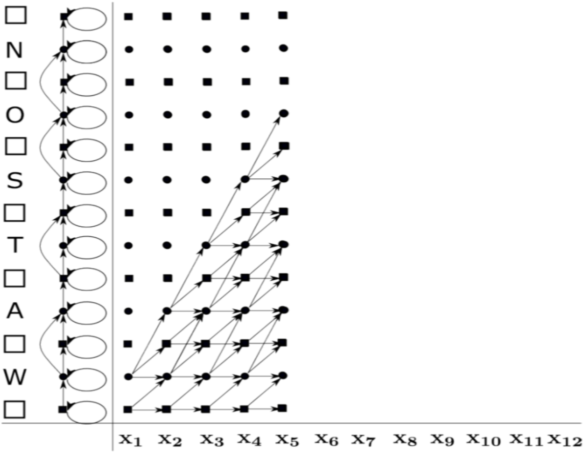
The CTC Forward-Backward Algorithm



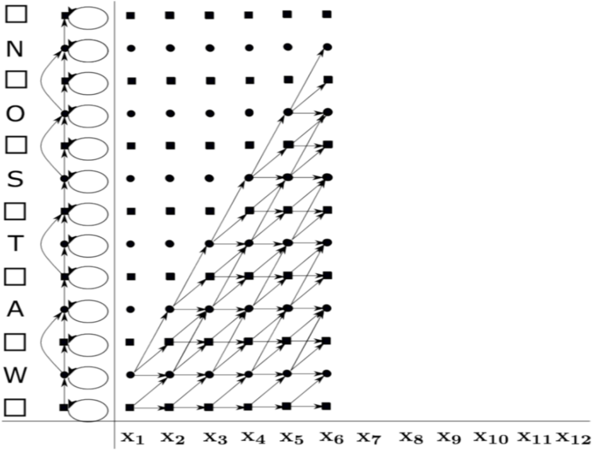
The CTC Forward-Backward Algorithm



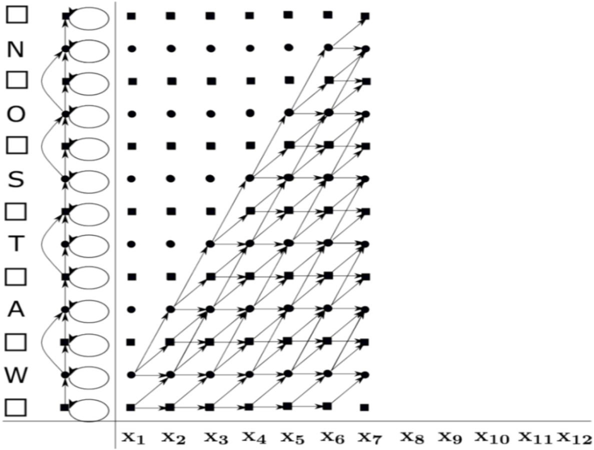
The CTC Forward-Backward Algorithm



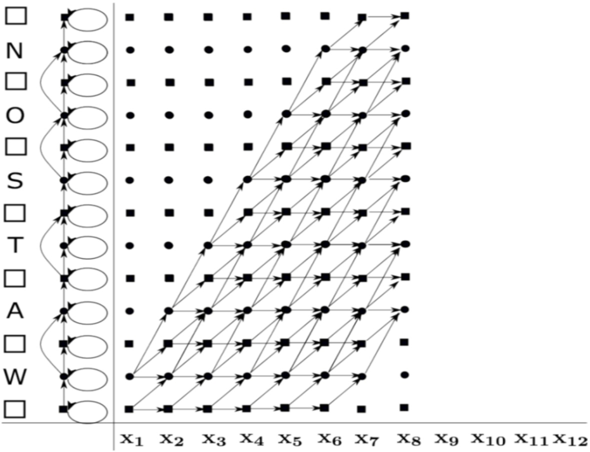
The CTC Forward-Backward Algorithm



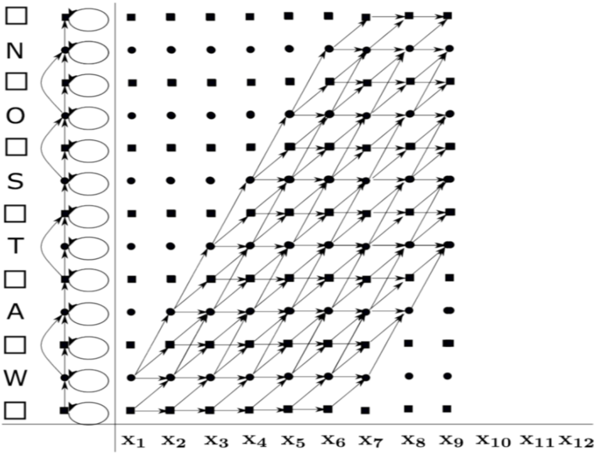
The CTC Forward-Backward Algorithm



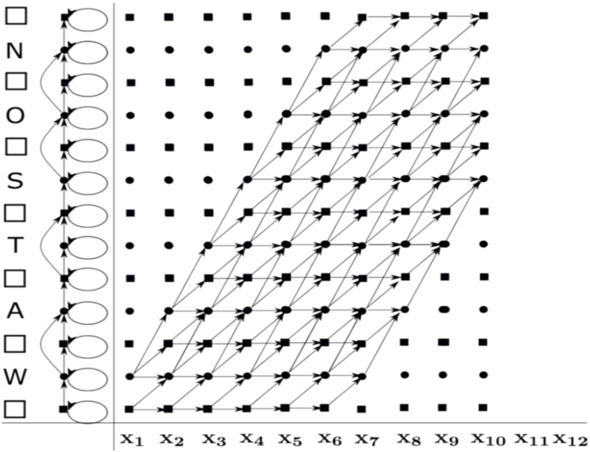
The CTC Forward-Backward Algorithm



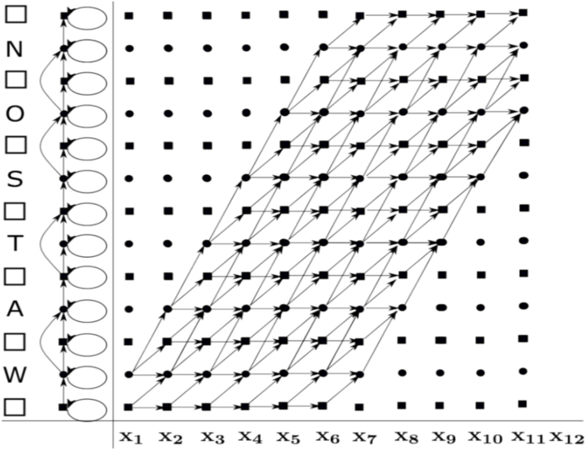
The CTC Forward-Backward Algorithm



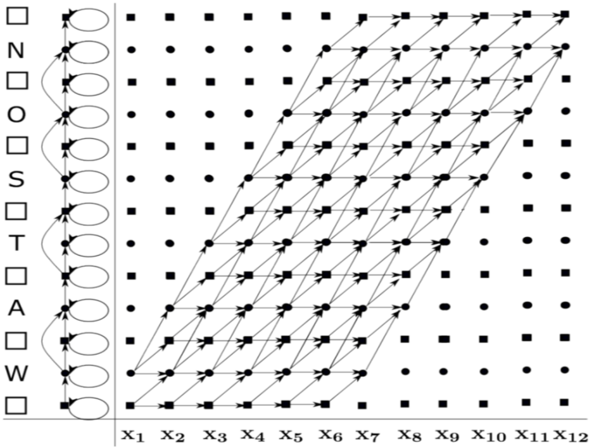
The CTC Forward-Backward Algorithm



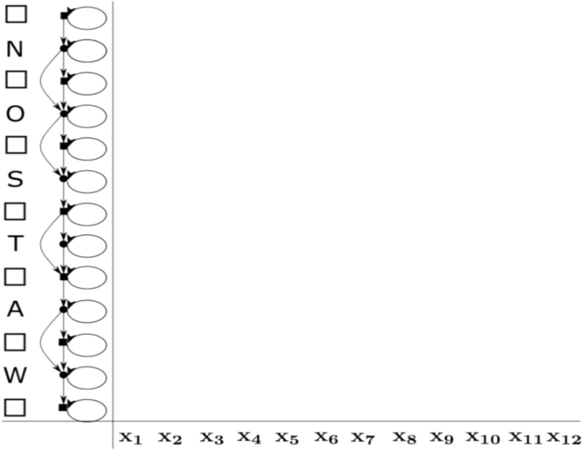
The CTC Forward-Backward Algorithm



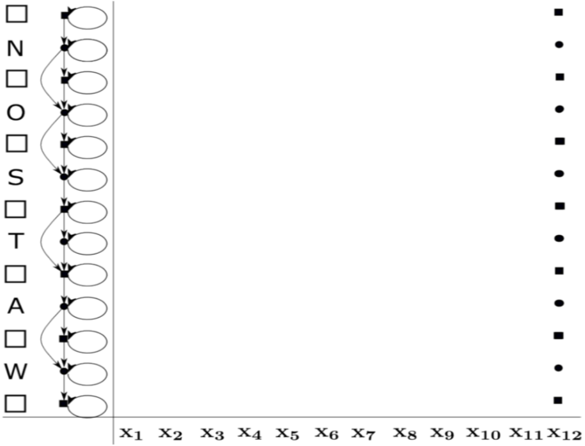
The CTC Forward-Backward Algorithm



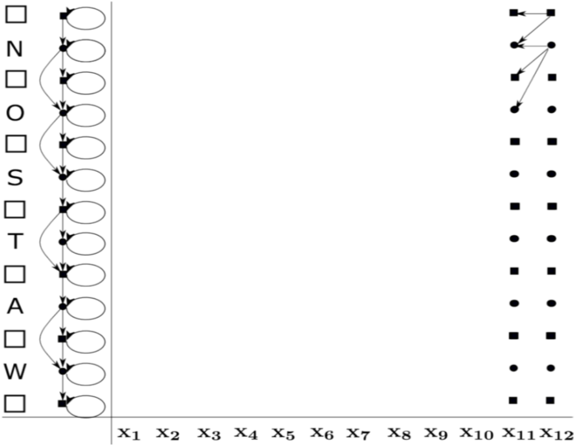
The CTC Forward-Backward Algorithm



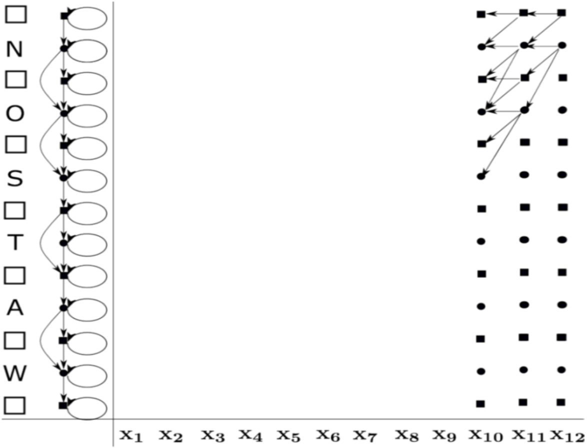
The CTC Forward-Backward Algorithm



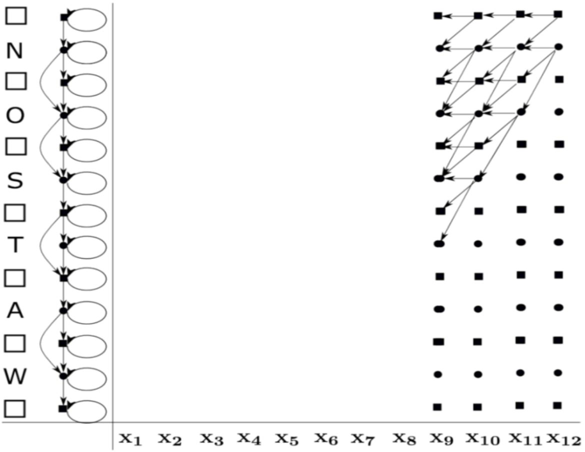
The CTC Forward-Backward Algorithm



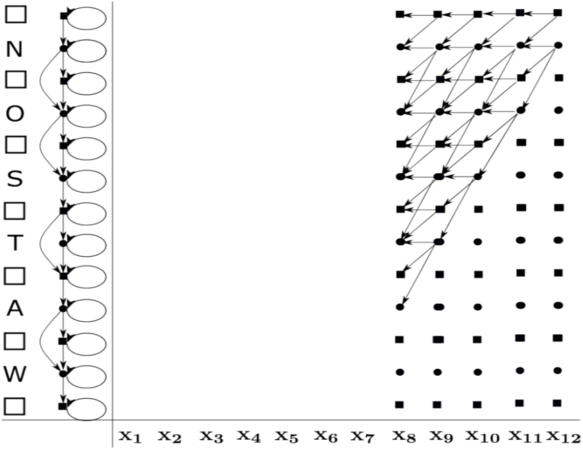
The CTC Forward-Backward Algorithm



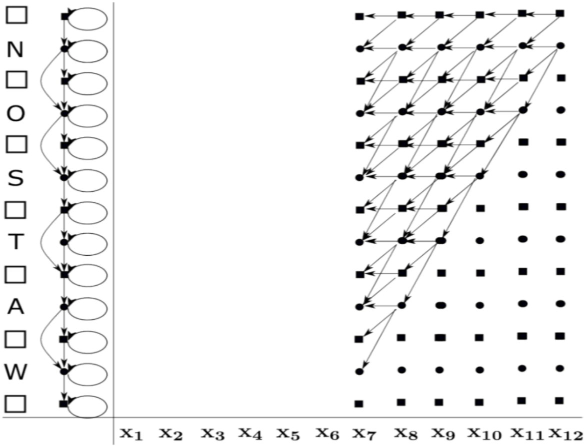
The CTC Forward-Backward Algorithm



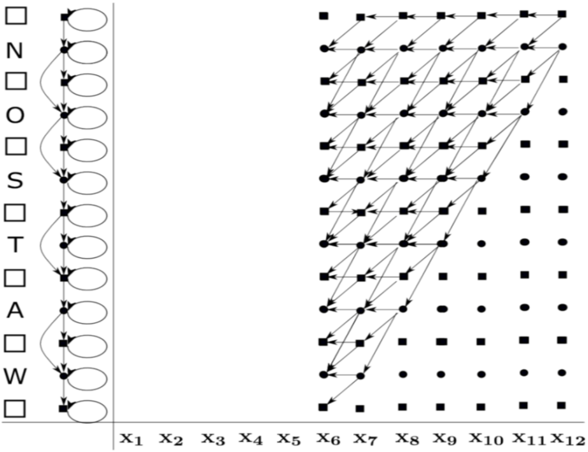
The CTC Forward-Backward Algorithm



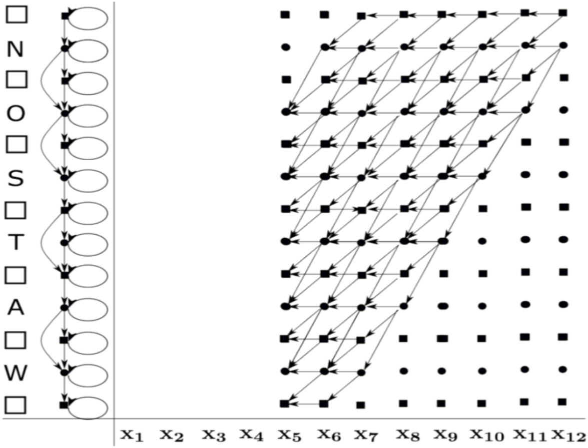
The CTC Forward-Backward Algorithm



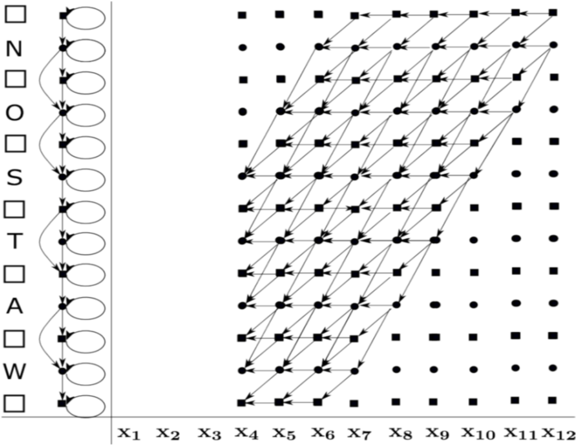
The CTC Forward-Backward Algorithm



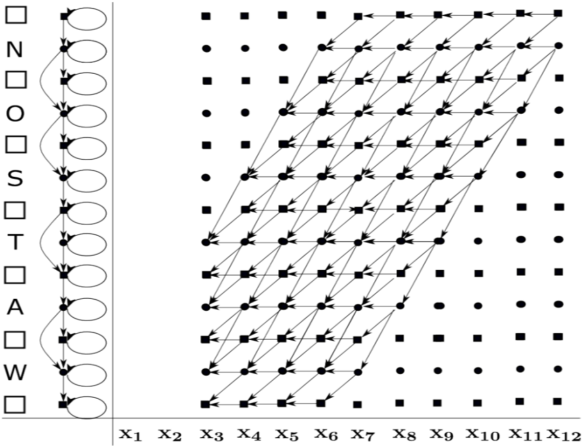
The CTC Forward-Backward Algorithm



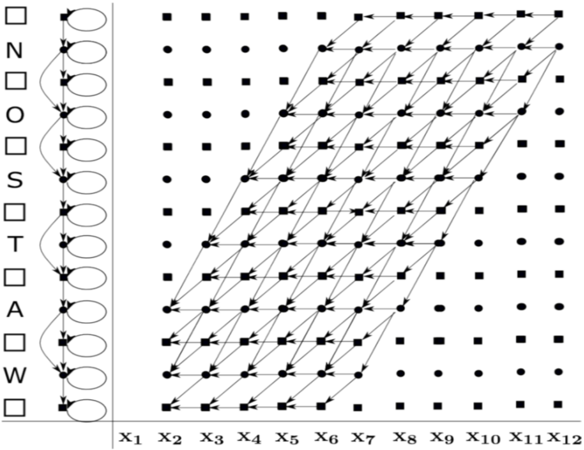
The CTC Forward-Backward Algorithm



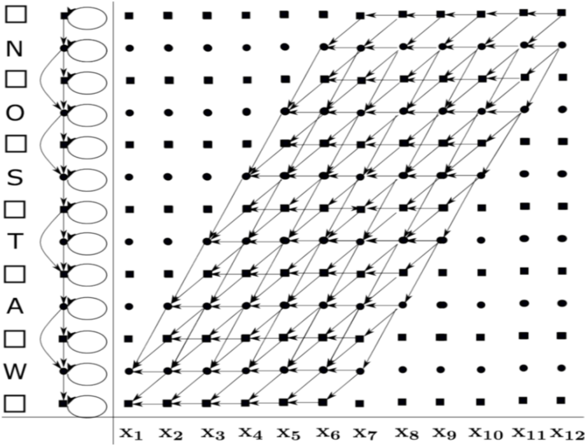
The CTC Forward-Backward Algorithm



The CTC Forward-Backward Algorithm



The CTC Forward-Backward Algorithm



The CTC Forward-Backward Algorithm

- Forward Computation: $\alpha_t(s) = P(x_1 \cdots x_t, \pi_t = s|\lambda)$
  - Initialization

$$\alpha_1(1) = y^1_{\Box}, \quad \alpha_1(2) = y^1_{z_1}, \quad \alpha_1(s) = 0, \ \forall s > 2$$

  - Recursion

$$\alpha_t(s) = \begin{cases} [\alpha_{t-1}(s) + \alpha_{t-1}(s-1)]y^t_{z'_s}, & \text{if } \ z'_s = \Box \ \text{ or } \ z'_{s-2} = z'_s. \\ [\alpha_{t-1}(s) + \alpha_{t-1}(s-1) + \alpha_{t-1}(s-2)]y^t_{z'_s}, & \text{otherwise} \end{cases}$$

  - Termination

$$P(Z|X) = \alpha_{\mathsf{T}}(|Z'|) + \alpha_{\mathsf{T}}(|Z'| - 1)$$

- Backward Computation: $\beta_t(s) = P(x_t \cdots x_{\mathsf{T}}|\pi_t = s, \lambda)$
  - Initialization

$$\beta_{\mathsf{T}}(|Z'|) = y^{\mathsf{T}}_{\Box}, \quad \beta_{\mathsf{T}}(|Z'| - 1) = y^{\mathsf{T}}_{z_{|Z|}}, \quad \alpha_{\mathsf{T}}(s) = 0, \ \forall s < |Z'| - 1$$

  - Recursion

$$\beta_t(s) = \begin{cases} [\beta_{t+1}(s) + \beta_{t+1}(s+1)]y^t_{z'_s}, & \text{if } \ z'_s = \Box \ \text{ or } \ z'_{s+2} = z'_s. \\ [\beta_{t+1}(s) + \alpha_{t+1}(s+1) + \alpha_{t+1}(s+2)]y^t_{z'_s}, & \text{otherwise} \end{cases}$$

CTC Maximum Likelihood Optimization

Objective function

$$\mathcal{L}_{\mathsf{CTC}} = \log P_\lambda(Z|X) \quad \text{where} \quad P(Z|X) = \sum_{s=1}^{|Z'|} \frac{\alpha_t(s)\beta_t(s)}{y^t_{z'_s}}$$

Gradient of $\mathcal{L}_{\mathsf{CTC}}$ with respect to the **unnormalized outputs** u^t_k of the network (a.k.a. the input to the softmax function):

$$\frac{\partial \mathcal{L}_{\mathsf{CTC}}}{\partial u^t_k} = y^t_k - \frac{1}{y^t_k P(Z|X)} \sum_{s \in \mathsf{lab}(Z,k)} \alpha_t(s)\beta_t(s) = y^t_k - \gamma^t_k$$

where

$$\gamma^t_k \triangleq P(s_t = k|Z', \lambda) = \frac{1}{y^t_k P(Z|X)} \sum_{s \in \mathsf{lab}(Z',k)} \alpha_t(s)\beta_t(s)$$

Recall for cross-entropy objective function, the gradient with respect to the input to the softmax:

$$\frac{\partial \mathcal{L}_{\mathsf{CE}}}{\partial u^t_k} = y^t_k - \bar{y}^t_k$$

- **soft labels** vs. **hard labels**

CTC Decoding

Alex Graves mentioned two decoding strategy in his seminal CTC paper

- Best path decoding

$$h(X) \approx \boldsymbol{\mathcal{B}}(\pi^*)$$

  where

$$\pi^* = \operatorname*{argmax}_{\pi \in N^t} p(\pi|X)$$

  simply concatenate the most active outputs at each time-step, not guaranteed to find the most probable labeling

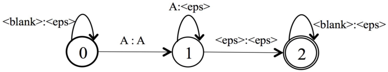
- Prefix search decoding with beam search
  - works better practically
  - may fail in some cases

*A. Graves, S. Fernandez, F. Gomez and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks", ICML, 2006

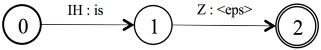
CTC WFST-based Decoding

Weighted Finite State Transducers (WFSTs)

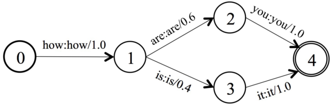
- Token T



- Lexion L



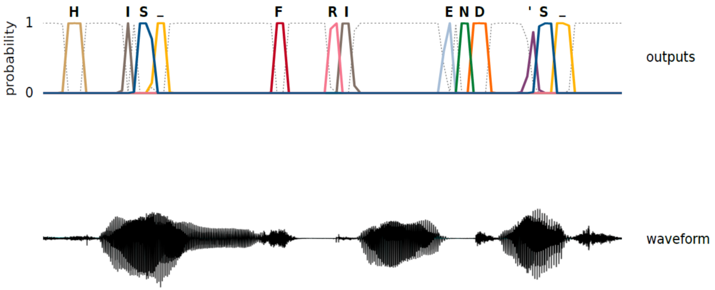
- Language G



- Search graph

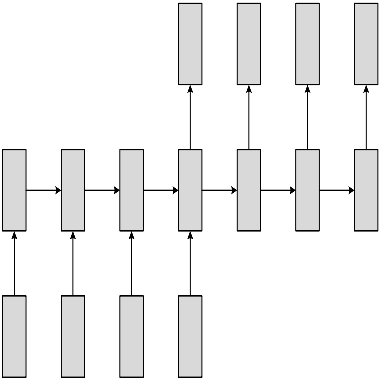
$$S = T \circ \min(\det(L \circ G))$$

CTC Output Behavior



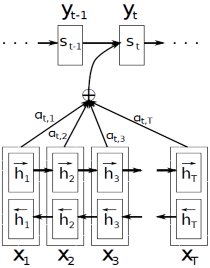
*A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks", ICML, 2014. Adapted.

Encoder-Decoder Architectures



Many-to-many sequence mapping.

Attention Mechanisms



- Score function:

$$e_{ti} = \mathsf{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i)$$

- Attention weights:

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{j=1}^{T_x} \exp(e_{tj})}$$

- Context vector:

$$\boldsymbol{c}_t = \sum_{i=1}^{T_x} \alpha_{ti} \boldsymbol{h}_i$$

*D. Bahdanau, K. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR, 2015.

Some Attention Functions

- Dot-Product-Attention

$$\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i) = \boldsymbol{s}_{t-1}^{\mathsf{T}} \mathbf{W} \boldsymbol{h}_i$$

- Additive-Attention

$$\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i) = \boldsymbol{v}^{\mathsf{T}} \tanh(\mathbf{W} \boldsymbol{s}_{t-1} + \mathbf{U} \boldsymbol{h}_i + \boldsymbol{b})$$

- Location-Based-Attention

$$\begin{aligned} \mathbf{F}_t &= \mathbf{K} * \boldsymbol{\alpha}_{t-1} \\ \text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i) &= \boldsymbol{v}^{\mathsf{T}} \tanh(\mathbf{W} \boldsymbol{s}_{t-1} + \mathbf{U} \boldsymbol{h}_i + \mathbf{V} \boldsymbol{F}_{t,i} + \boldsymbol{b}) \end{aligned}$$

- Multi-Head-Attention

Decoding in the Encoder-Decoder Architecture

- Decoder as a sequence generation model:
  - At each time stamp, the decoder generates a probability distribution over the vocabulary

$$P(z_t | z_{t-1}, \cdots, z_1; x_1, x_2, \cdots, x_{\mathsf{T}})$$

  - Draw a word from the vocabulary according to the distribution
  - Feed it as input to the next time stamp
  - Repeat until an $<$EOS$>$ shows up.
- The goal is to generate the most likely output word sequence
- Solutions
  - Simply picking the most likely word at each time stamp is suboptimal
  - Keep multiple hypotheses and conduct the beam search

Data Augmentation by Label Preserving Transformations

- Artificially augment the training set using replicas of training samples under certain transformations.
- Make neural networks invariant to such transformations.
- Helpful when training data is limited.
- Some commonly-used approaches
    - perturbation of speaking rate
    - perturbation of vocal tract length
    - voice conversion in some designated feature space by stochastic mapping
    - multi-style training by adding noise

Adaptation of Acoustic Models – Transfer Learning

- A classifier $P_\theta(Y|X)$

  A training population distribution $P_s(X)$

  A test population distribution $P_t(X)$

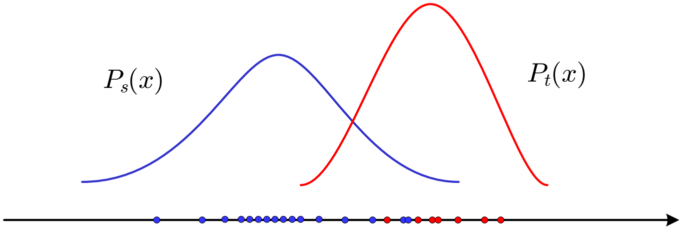
  $X \in \mathbf{R}^n$, $Y \in \mathbf{R}^m$

- $P_\theta(Y|X)$ is learned from training data under distribution $P_s(X)$

- Same $P_\theta(Y|X)$ is used for classification on test data under distribution $P_t(X)$

- What happens if $P_s(X) \neq P_t(X)$?

Distribution Mismatch In Transfer Learning



Why Adaptation Is Needed In Acoustic Modeling

- Speech signals are affected by a variety of variabilities
  - speaker
  - environment
  - channel
  - ......
- An important issue to deal with in acoustic modeling
  - a test speech signal may come from a sparse region of the training distribution, which may give rise to performance degradation
  - adaptation or adaptive training is constantly pursued to mitigate the distribution mismatch

Adaptation of DNN-HMMs

- What did we do in GMM-HMMs?
  - elegant mathematical models (MLLR, fMLLR, MAP, eigenVoice, ...)
  - exploit the generative structure of GMM-HMM for parameter tying
- What's the challenge of adapting DNN-HMMs?
  - substantial number of parameters $\rightarrow$ data sparsity seems to always be the issue for DNN adaptation
  - lack of a generative structure for parameter tying
  - unsupervised adaptation, which is preferred in practice, makes it even harder due to the strong discriminative nature of DNNs.
  - catastrophic forgetting

Some Commonly-used Techniques for DNN Adaptation

- Use speaker-adapted input features (e.g. fMLLR, VTL-warped logMEL)
- Fine-tune the whole network with a small learning rate
- Fine-tune or retrain a selected subset of the network parameters
  - input/output/hidden layer(s)
  - SVD factorization of the output layer ($m > n$, $n \gg k$)

$$\mathbf{W}_{\mathsf{m\times n}} = \mathbf{U}_{\mathsf{m\times m}}\mathbf{\Sigma}_{\mathsf{m\times n}}\mathbf{V}^{\mathsf{T}}_{\mathsf{n\times n}} \approx \mathbf{U}_{\mathsf{m\times k}}\mathbf{\Sigma}_{\mathsf{k\times k}}\mathbf{V}^{\mathsf{T}}_{\mathsf{n\times k}} = \mathbf{U}_{\mathsf{m\times k}}\mathbf{N}_{\mathsf{k\times n}}$$

$$\tilde{\mathbf{W}}_{\mathsf{m\times n}} = \mathbf{U}_{\mathsf{m\times k}}\mathbf{S}_{\mathsf{k\times k}}\mathbf{N}_{\mathsf{k\times n}}$$

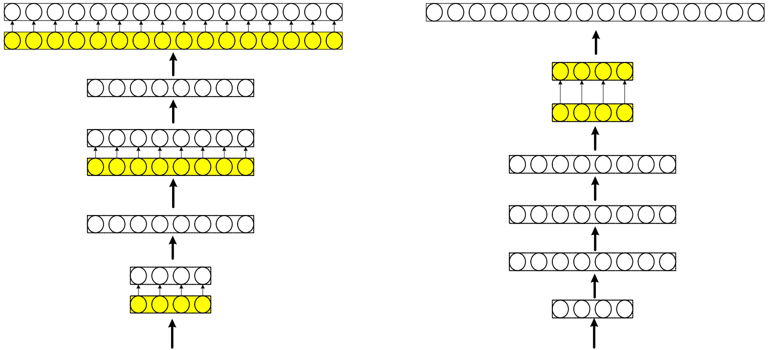
- Learning hidden unit contributions (LHUC)

$$a_i^{(\mathsf{l})} = \sum_j w_{ij}^{(\mathsf{l})} z_j^{(\mathsf{l}-1)} + b_i^{(\mathsf{l})}, \qquad z_i^{(\mathsf{l})} = \gamma_i^{(\mathsf{l})} \cdot \sigma\left(a_i^{(\mathsf{l})}\right)$$

  - motivated a family of parameterized activation functions
- Speaker-aware training based on i-vectors

$$x \mapsto [x, e]$$

Some Commonly-used Techniques for DNN Adaptation



Multilingual Acoustic Modeling

- Oftentimes, to build an ASR system, the acoustic resources for a particular language or a particular domain is limited.
- Universal acoustic representations can significantly help this situation.
  - mitigate sparse data issue
  - better performance
  - faster system turn-around
- Multilingual acoustic modeling
  - Learning feature representations of universal acoustic characteristics from numerous languages
  - deep learning is especially suitable for multilingual acoustic modeling

A Case Study of Multilingual Feature Extraction

- 24 languages under the IARPA Babel program
- Cantonese, Assamese, Bengali, Pashto, Turkish, Tagalog, Vietnamese, Haitian Creole, Swahili, Lao, Tamil, Kurmanji Kurdish, Zulu, Tok Pisin, Cebuano, Kazakh, Telugu, Lithuanian, Amharic, Dholuo, Guarani, Igbo, Javanese, Mongolian.
- 40-70 hours of labeled speech data from each language.