Semi-supervised Maximum Mutual Information Training of Deep Neural Network Acoustic Models

Vimal Manohar[1], Daniel Povey [12], Sanjeev Khudanpur[12]

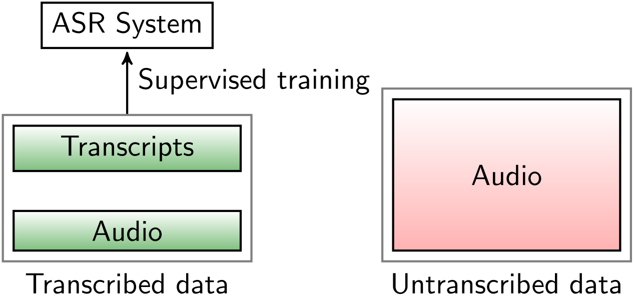
[1]Center for Language and Speech Processing
[2]Human Language Technology Center of Excellence
The Johns Hopkins University, Baltimore, MD 21218, USA

CLSP Seminar, 2015

Outline

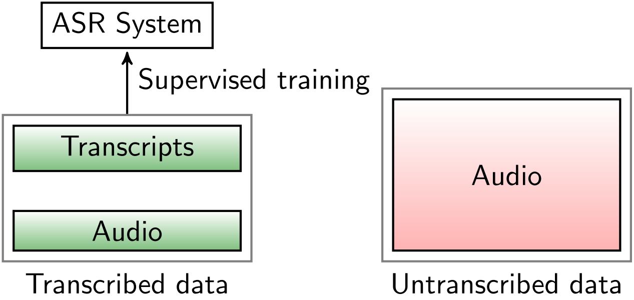
Outline

Speech Data



- Audio: Spectral features extracted from wav files
- Transcription: Word sequences (subtitles)

Speech Data



- Audio: Spectral features extracted from wav files
- Transcription: Word sequences (subtitles)
- Why do we want to use untranscribed data?

Outline

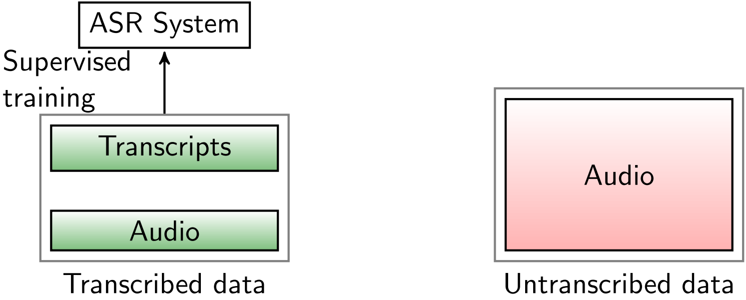
Unsupervised Approaches - Motivations

Why do we want to use untranscribed data?

- Availability of exponentially large amounts of untranscribed acoustic data
- Interests in speech recognition in low-resource languages
- Test data changes with time.

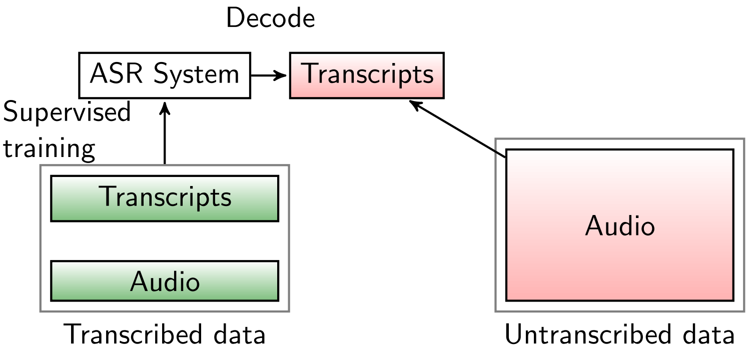
Unsupervised Approaches - Self-training

- Self-training: Use a seed model trained on transcribed data to transcribe the audio



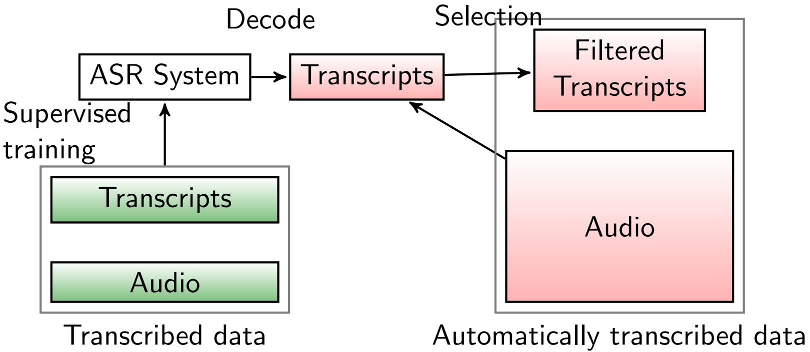
Unsupervised Approaches - Self-training

- Self-training: Use a seed model trained on transcribed data to transcribe the audio

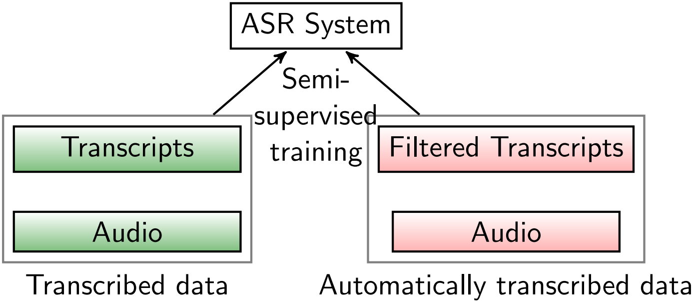


Unsupervised Approaches - Self-training

- Self-training: Use a seed model trained on transcribed data to transcribe the audio



Unsupervised Approaches - Self-training



Unsupervised Approaches - Self-training

- Data selection can involve:
  - Confidence-based filtering to select 'good' data
  - Confidence-based weighting
- Sentence-level scores, word-level scores, frame-level scores etc.

Unsupervised Approaches - Self-training

Problem

The decision for using the data (filtering or weighting) is done before training, and not incorporated in the training process.

Solution

Incorporate confidences into the objective function

Unsupervised Approaches - Self-training

Problem

The decision for using the data (filtering or weighting) is done before training, and not incorporated in the training process.

Solution

Incorporate confidences into the objective function

Outline

Acoustic Model

Basic Speech Recognition problem:

- Given an acoustic feature sequence O, find the transcript (word sequence) W that best describes it.

$$\begin{aligned}\hat{W} &= \arg\max_{W \in \mathcal{H}} P(W \mid O) \\ &= \arg\max_{W \in \mathcal{H}} P_A(O \mid W) P_L(W) \qquad (1)\end{aligned}$$

where
$P_A(O \mid W)$ is called the Acoustic Model,
$P_L(W)$ is called the Language Model

Acoustic Model

- Labelled training data $(\mathcal{D}_{\mathcal{L}})$: L utterances with
  - Acoustic features - $\mathbf{O}^{(1)}, \mathbf{O}^{(2)} \ldots \mathbf{O}^{(L)}$
  - Corresponding transcripts - $W^{(1)}, W^{(2)} \ldots W^{(L)}$
- Maximum Likelihood Estimation

$$\mathcal{F}_{\mathrm{ML}}(\lambda) = \sum_{r=1}^{L} \log P_A(\mathbf{O}^{(r)} \mid W^{(r)}; \lambda) \tag{2}$$

- Maximum Mutual Information Estimation

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \mathbb{I}(\mathbf{O} : W) \tag{3}$$

Maximum Mutual Information (MMI)

$$
\begin{aligned}
\mathcal{F}_{\mathrm{MMI}}(\lambda) &= \mathbb{I}(\mathbf{O}; W) \\
&= \frac{1}{L} \sum_{r=1}^{L} \log \frac{P(\mathbf{O}^{(r)}, W^{(r)})}{P(\mathbf{O}^{(r)}) P(W^{(r)})} \quad \text{(Empirical MI)} \\
&\propto \frac{1}{L} \sum_{r=1}^{L} \log P(W^{(r)} \mid \mathbf{O}^{(r)}) \quad \text{(Conditional Likelihood)} \\
&\propto \frac{1}{L} \sum_{r=1}^{L} \log \frac{P_A(\mathbf{O}^{(r)} \mid W^{(r)})}{\sum_{W'} P_A(\mathbf{O}^{(r)} \mid W') P_L(W')}
\end{aligned}
$$

Log-likelihood under the reference transcript − Log-likelihood under all possible transcripts.

Discriminative Training

$$\begin{array}{c} \text{Log-likelihood under} \\ \text{the reference transcript} \end{array} - \begin{array}{c} \text{Log-likelihood under} \\ \text{all possible transcripts.} \end{array}$$

Problem

- Standard discriminative training is very sensitive to the accuracy of the transcripts (Mathias et al. [2005]).
- So self-training methods do not work very well with discriminative training.

Solution

As before, we modify the objective function.

Discriminative Training

- Discriminative training tries to make the model maximally discriminative of the reference transcript against the competing hypotheses.

Problem

- Standard discriminative training is very sensitive to the accuracy of the transcripts (Mathias et al. [2005]).
- So self-training methods do not work very well with discriminative training.

Solution

As before, we modify the objective function.

Discriminative Training

- Discriminative training tries to make the model maximally discriminative of the reference transcript against the competing hypotheses.

Problem

- Standard discriminative training is very sensitive to the accuracy of the transcripts (Mathias et al. [2005]).
- So self-training methods do not work very well with discriminative training.

Solution

As before, we modify the objective function.

Discriminative Training

- Discriminative training tries to make the model maximally discriminative of the reference transcript against the competing hypotheses.

Problem

- Standard discriminative training is very sensitive to the accuracy of the transcripts (Mathias et al. [2005]).
- So self-training methods do not work very well with discriminative training.

Solution

As before, we modify the objective function.

Maximum Mutual Information (MMI)

- Transcribed data – Conditional Likelihood

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \frac{1}{L}\sum_{r=1}^{L} \log \mathbb{P}(W^{(r)} \mid \mathbf{O}^{(r)})$$

Maximum Mutual Information (MMI)

- Transcribed data – Conditional Likelihood

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \frac{1}{L}\sum_{r=1}^{L} \log \mathbb{P}(W^{(r)} \mid \mathbf{O}^{(r)})$$

- Automatically transcribed data – Conditional Likelihood

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \frac{1}{U}\sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}) \log \mathbb{P}(W^{(r)} \mid \mathbf{O}^{(r)})$$

$$= -\sum_{r=1}^{U} \mathbb{H}(W \mid \mathbf{O}^{(r)}; \lambda)$$

Maximum Mutual Information (MMI)

- Transcribed data – Conditional Likelihood

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \frac{1}{L}\sum_{r=1}^{L} \log \mathbb{P}(W^{(r)} \mid \mathbf{O}^{(r)})$$

- Untranscribed data – Negative Conditional Entropy

$$\mathcal{F}_{\mathrm{NCE}}(\lambda) = \frac{1}{U}\sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}) \log \mathbb{P}(W \mid \mathbf{O}^{(r)})$$

$$= -\sum_{r=1}^{U} \mathbb{H}(W \mid \mathbf{O}^{(r)}; \lambda)$$

Lattice Entropy

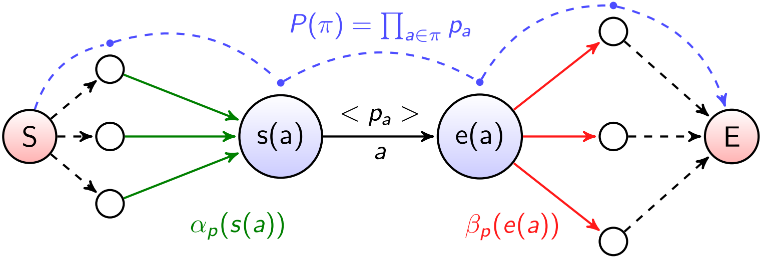
- Take weighted average over all possible hypotheses in the lattice

$$\mathcal{F}_{\mathrm{NCE}}(\lambda) = \sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda) \log \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda)$$
$$= -\sum_{r=1}^{U} \mathbb{H}(W \mid \mathbf{O}^{(r)}; \lambda)$$

- This is exactly minimizing the lattice entropy.
- Minimizing lattices entropy makes the model more confident of one of the hypotheses.

Lattice Entropy – Forward-Backward

- Forward-backward algorithm[1] over the lattice using WFSTs
- $Z = \sum_{\pi \in \mathcal{L}} P(\pi)$



$$\alpha_p(s) = \sum_{s'} \alpha_p(s') p_{s's} \qquad \beta_p(s) = \sum_{s'} \beta_p(s') p_{s's}$$

[1] Li and Eisner [2009], Huang [2012]

Lattice Entropy – WFST

$$\mathcal{F}_{\mathrm{NCE}}(\lambda) = \sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda) \log \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda)$$

Lattice Entropy – WFST

$$\mathcal{F}_{\mathrm{NCE}}(\lambda) = \sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda) \log \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda)$$

$$H_{\mathcal{L}} = -\sum_{\pi \in \mathcal{L}} P(\pi) \log P(\pi)$$

Lattice Entropy – WFST

$$\mathcal{F}_{\mathrm{NCE}}(\lambda) = \sum_{r=1}^{U} \sum_{W} \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda) \log \mathbb{P}(W \mid \mathbf{O}^{(r)}; \lambda)$$

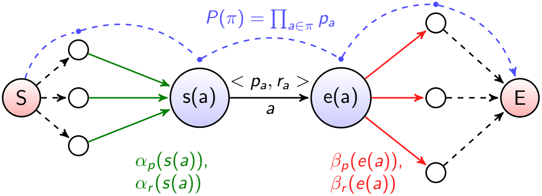
$$H_{\mathcal{L}} = -\sum_{\pi \in \mathcal{L}} P(\pi) \log P(\pi)$$

- Do forward-backward, but with a new semiring.
- Replace p_a with a pair $< p_a, p_a \log p_a >$.

Operation	Value
$< p_1, p_1 \log p_1 > \otimes < p_2, p_2 \log p_2 >$	$< p_1 p_2, p_1 p_2 \log p_2 + p_2 p_1 \log p_1 >$
$< p_1, p_1 \log p_1 > \oplus < p_2, p_2 \log p_2 >$	$< p_1 + p_2, p_1 \log p_1 + p_2 \log p_2 >$
Zero()	$< 0, 0 >$
One()	$< 1, 0 >$

Lattice Entropy – WFST

$$H_{\mathcal{L}} = -\sum_{\pi \in \mathcal{L}} P(\pi) \log P(\pi)$$



Operation	Value
$< p_1, p_1 \log p_1 > \otimes < p_2, p_2 \log p_2 >$	$< p_1 p_2, p_1 p_2 \log p_2 + p_2 p_1 \log p_1 >$
$< p_1, p_1 \log p_1 > \oplus < p_2, p_2 \log p_2 >$	$< p_1 + p_2, p_1 \log p_1 + p_2 \log p_2 >$
Zero()	$< 0, 0 >$
One()	$< 1, 0 >$

Lattice Entropy – Gradients

$$H_{\mathcal{L}} = -\sum_{\pi \in \mathcal{L}} P(\pi) \log P(\pi)$$

$$\nabla H_{\mathcal{L}} = ??$$

- We have used a Forward-Backward to compute the objective function.
- But we need derivatives of the objective function.
- Another pass over lattice.
- The derivatives are aggregated over the arcs for each context-dependent state and then backpropagated through the DNN.

Lattice Entropy – Gradients

$$H_{\mathcal{L}} = -\sum_{\pi \in \mathcal{L}} P(\pi) \log P(\pi)$$

$$\nabla H_{\mathcal{L}} = ??$$

- We have used a Forward-Backward to compute the objective function.
- But we need derivatives of the objective function.
- Another pass over lattice.
- The derivatives are aggregated over the arcs for each context-dependent state and then backpropagated through the DNN.

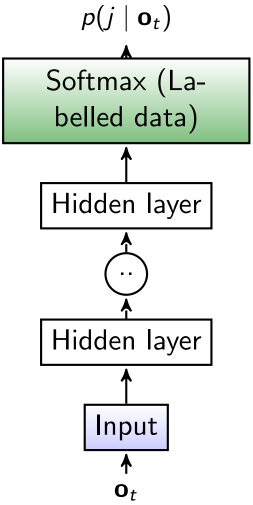
Outline

1. Introduction
2. Unsupervised Training
3. Discriminative training
   - Maximum Mutual Information (MMI)
   - Lattice Entropy
   - Lattice computations
4. Deep Neural Network
   - Baseline
   - Multilingual-inspired architecture
   - DNN Priors
5. Experiments

Deep Neural Network - CE Supervised training

- $\mathcal{D}_\mathcal{L} = \{(\mathbf{O}^{(r)}, W^{(r)})_{r=1}^{L}\}$
- Word sequences $W^{(r)}$ are converted to frame-alignment using Viterbi alignment
- For each frame t, we have a context-dependent phone j
- $\mathcal{D}_\mathcal{L} = \{(\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_L\}\}$

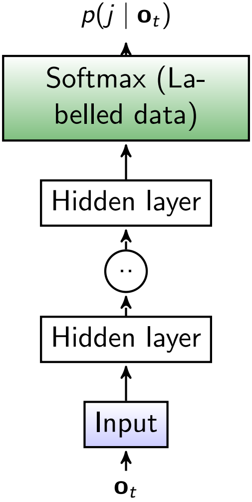
$$\mathcal{F}_{\mathrm{CE}} = -\sum_t \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t) \tag{4}$$



Deep Neural Network - CE Supervised training

- $\mathcal{D}_{\mathcal{L}} = \{(\mathbf{O}^{(r)}, W^{(r)})_{r=1}^{L}\}$
- Word sequences $W^{(r)}$ are converted to frame-alignment using Viterbi alignment
- For each frame t, we have a context-dependent phone j
- $\mathcal{D}_{\mathcal{L}} = \{(\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_L\}\}$

$$\mathcal{F}_{\mathrm{CE}} = -\sum_{t}\sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t) \tag{4}$$

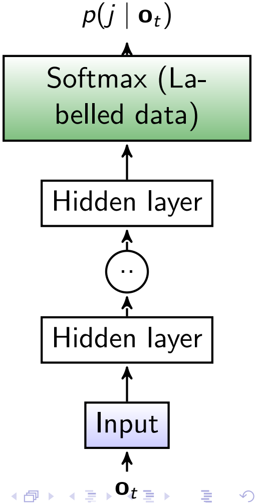


Deep Neural Network - CE Self-training

Viterbi alignment

$$\mathcal{D}_{\mathcal{L}} = \{(\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_L\}\}$$

$$\mathcal{F}_{\mathrm{CE}} = -\sum_{t=1}^{N_L} \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t)$$

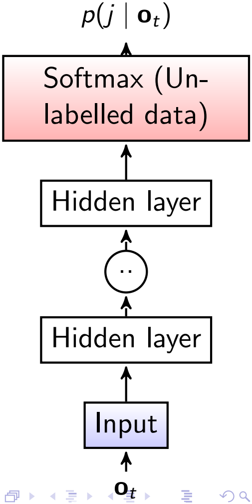


Deep Neural Network - CE Self-training

Viterbi decode

$$\mathcal{D}_{\mathcal{U}} = \left\{ (\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_U\} \cap \left\{ t : \frac{p_a}{Z} \geq 0.7 \right\} \right\}$$

$$\mathcal{F}_{\mathrm{CE}} = -\sum_{t=1}^{N_U} \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t)$$

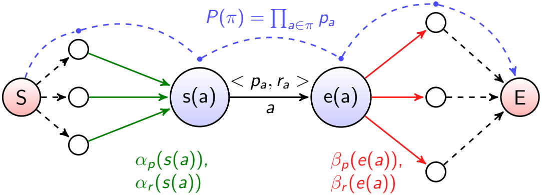


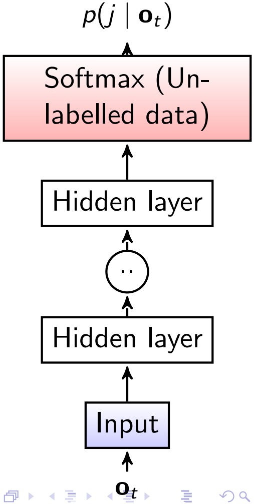
Deep Neural Network - CE Self-training

Viterbi decode

$$\mathcal{D}_{\mathcal{U}} = \left\{ (\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_U\} \cap \left\{ t : \frac{p_a}{Z} \geq 0.7 \right\} \right\}$$

$$\mathcal{F}_{\text{CE}} = -\sum_{t=1}^{N_U} \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t)$$



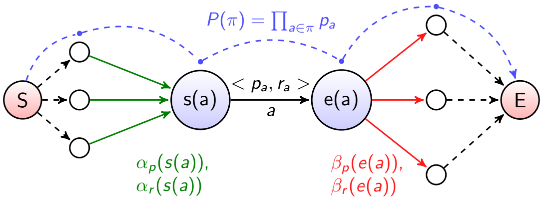


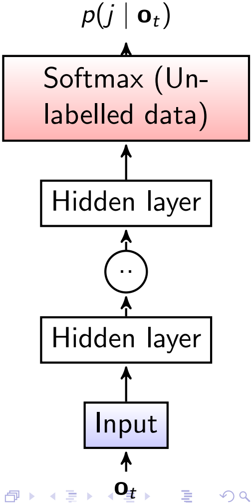
Deep Neural Network - CE Self-training

Viterbi decode

$$\mathcal{D}_{\mathcal{U}} = \left\{ (\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_U\} \cap \left\{ t : \frac{p_a}{Z} \geq 0.7 \right\} \right\}$$

$$\mathcal{F}_{\text{CE}} = -\sum_{t=1}^{N_U} \frac{p_a}{Z} \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t)$$



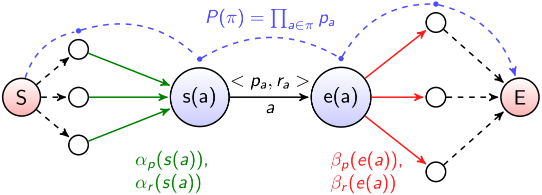


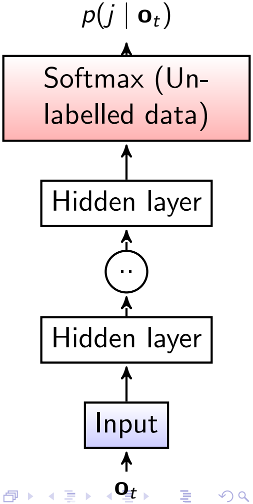
Deep Neural Network - CE Self-training

Viterbi decode

$$\mathcal{D}_{\mathcal{U}} = \left\{ (\mathbf{o}_t, \mathbf{z}_t) : t \in \{1 \cdots N_U\} \cap \left\{ t : \frac{p_a}{Z} \geq 0.7 \right\} \right\}$$

$$\mathcal{F}_{\mathrm{CE}} = -\sum_{t=1}^{N_U} \frac{p_a}{Z} \sum_{j=1}^{K} z_{tj} \log p(j \mid \mathbf{o}_t)$$





Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

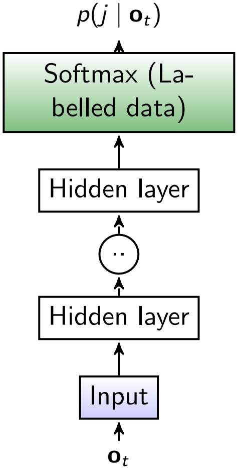


Figure: Baseline Architecture

[4]Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

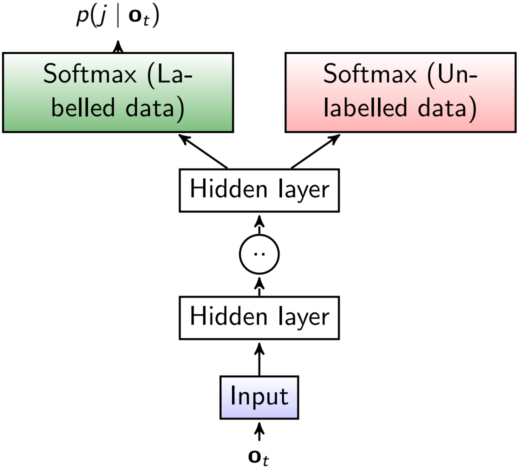


Figure: During an iteration

[4]Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

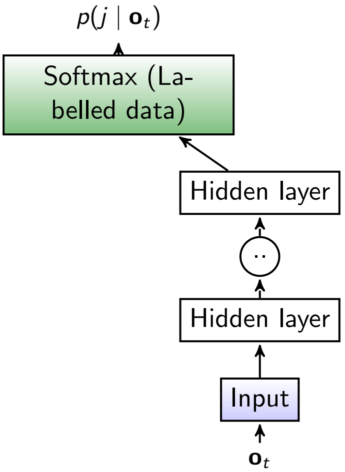


Figure: After the iteration

[4] Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

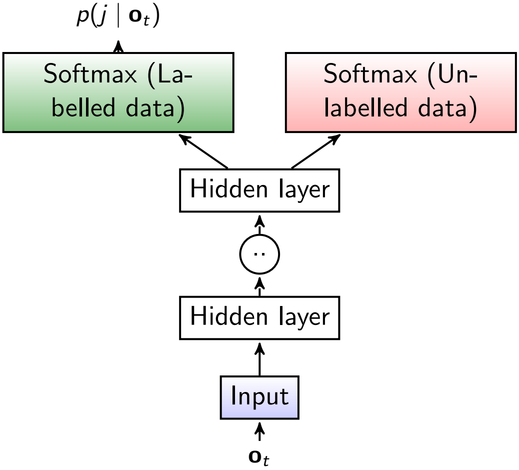


Figure: Before the next iteration

[4]Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

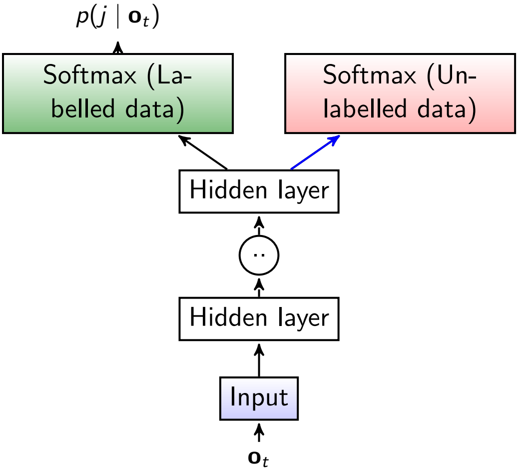


Figure: Scaling down gradients

[4]Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

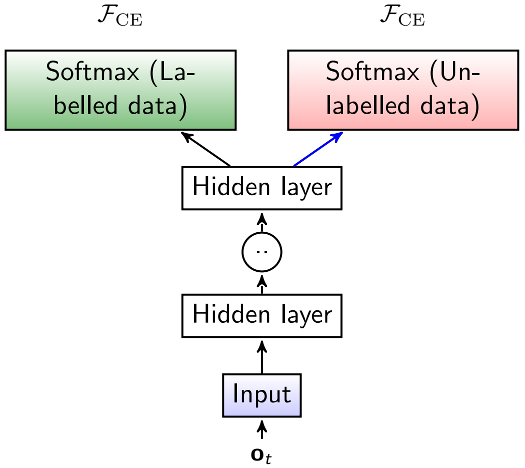


Figure: Different objectives

[4] Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

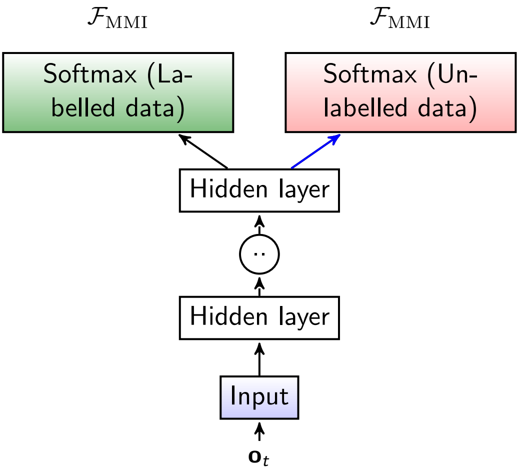


Figure: Different objectives

[4] Heigold et al. [2013]

Deep Neural Network

- Multi-task architecture[2].
- Final softmax layer corresponding to unlabelled data is **discarded** at the end of every iteration.
- Scaling down gradients
- Different objective functions

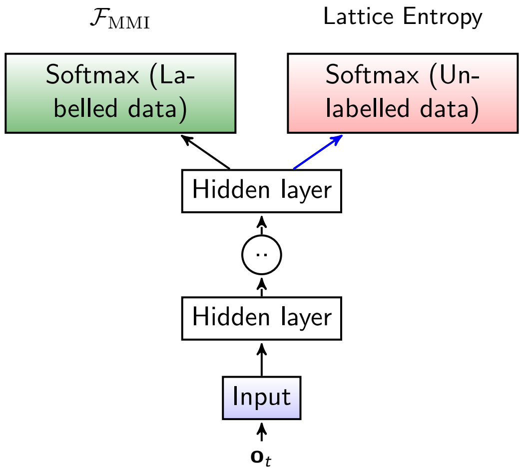


Figure: Different objectives

[4]Heigold et al. [2013]

Multi-task architecture - Advantages

- Reduces the tendency to learn degenerate distributions – Since lattice entropy can arbitrarily be made zero
- Force the neural network to also be good on the supervised MMI objective.

Deep Neural Network - Priors

- The DNN outputs are converted into likelihoods by dividing by a "prior" term:

$$p(\mathbf{o}_i \mid j) = \frac{p(j \mid \mathbf{o}_i)}{P(j)}$$

- The usual way of computing priors is using HMM-GMM reference alignments.
- The prior $P(j)$ is computed here by marginalizing the DNN outputs over the training data.

$$P(j) = \frac{1}{N} \sum_{i=1}^{N} p(j \mid \mathbf{o}_i).$$

- The highest change in priors is observed for the silence pdfs.

Outline

Experimental Setup

- Corpora:
  - Fisher English - 100 hours labelled data + 250 hours unlabelled
  - Babel - 4 languages in *limitedLP* condition (10 hours labelled data + 60 hours unlabelled data)
- Language model trained only on the labelled data.
- Acoustic model
  - DNN with p-norm non-linearities
  - fMLLR speaker-adapted LDA+MLLT features as input
- WER improvements also found to hold with using Time-delay neural networks (TDNN) and i-vector adaptation systems
- Tried 3-gram, 2-gram and unigram LMs for decoding unlabelled data. 3-gram gave the best performance.

Results - Fisher English

Table: *WER (%) results on Fisher English*

Type	Objective: Sup	Objective: Unsup	*dev*	*test*
Supervised training (100hr)	*CE*	-	32.0	31.2
	sMBR[3]	-	29.6	28.5
Self-training (100hr + 250hr)	*CE*	*CE*[4]	32.5	—
	sMBR	*sMBR*	—	—
Self-training in Multitask architecture	*CE*	*CE*	30.5	29.8
	sMBR	*sMBR*	29.9	28.8
Semi-supervised in Multitask architecture	*sMBR*	*Lattice Entropy*	29.4	28.1
Oracle Supervised training (350hr)	*sMBR*	-	28.5	27.5

[3] sMBR = state Minimum Bayes Risk

[4] Vesely et al. [2013]

Results - Fisher English

Table: *WER (%) results on Fisher English*

Type	Objective: *Sup*	Objective: *Unsup*	*dev*	*test*
Supervised training (100hr)	*CE*	-	32.0	31.2
	sMBR[3]	-	29.6	28.5
Self-training (100hr + 250hr)	*CE*	*CE* [4]	32.5	—
	sMBR	*sMBR*	—	—
Self-training in Multitask architecture	*CE*	*CE*	30.5	29.8
	sMBR	*sMBR*	29.9	28.8
Semi-supervised in Multitask architecture	*sMBR*	*Lattice Entropy*	29.4	28.1
Oracle Supervised training (350hr)	*sMBR*	-	28.5	27.5

[3] sMBR = state Minimum Bayes Risk

[4] Vesely et al. [2013]

Results - Fisher English

Table: *WER (%) results on Fisher English*

Type	Objective: Sup	Objective: Unsup	*dev*	*test*
Supervised training (100hr)	*CE*	-	32.0	31.2
	sMBR[3]	-	29.6	28.5
Self-training (100hr + 250hr)	*CE*	*CE* [4]	32.5	—
	sMBR	*sMBR*	—	—
Self-training in Multitask architecture	*CE*	*CE*	30.5	29.8
	sMBR	*sMBR*	29.9	28.8
Semi-supervised in Multitask architecture	*sMBR*	*Lattice Entropy*	29.4	28.1
Oracle Supervised training (350hr)	*sMBR*	-	28.5	27.5

[3] sMBR = state Minimum Bayes Risk

[4] Vesely et al. [2013]

Results - Fisher English

Table: *WER (%) results on Fisher English*

Type	Objective		*dev*	*test*
	Sup	*Unsup*		
Supervised training (100hr)	*CE*	-	32.0	31.2
	sMBR[3]	-	29.6	28.5
Self-training (100hr + 250hr)	*CE*	*CE* [4]	32.5	—
	sMBR	*sMBR*	—	—
Self-training in Multitask architecture	*CE*	*CE*	30.5	29.8
	sMBR	*sMBR*	29.9	28.8
Semi-supervised in Multitask architecture	*sMBR*	*Lattice Entropy*	29.4	28.1
Oracle Supervised training (350hr)	*sMBR*	-	28.5	27.5

[3] sMBR = state Minimum Bayes Risk
[4] Vesely et al. [2013]

Results - Fisher English

Table: *WER (%) results on Fisher English*

Type	Objective: Sup	Objective: Unsup	dev	test
Supervised training (100hr)	*CE*	-	32.0	31.2
	sMBR[3]	-	29.6	28.5
Self-training (100hr + 250hr)	*CE*	*CE* [4]	32.5	—
	sMBR	*sMBR*	—	—
Self-training in Multitask architecture	*CE*	*CE*	30.5	29.8
	sMBR	*sMBR*	29.9	28.8
Semi-supervised in Multitask architecture	*sMBR*	*Lattice Entropy*	29.4	28.1
Oracle Supervised training (350hr)	*sMBR*	-	28.5	27.5

[3] sMBR = state Minimum Bayes Risk
[4] Vesely et al. [2013]

Results - Babel

Table: *WER (%) results on Babel – (10hr + 60hr)*

Language	Objective		*dev2h*	*dev10h*
	Sup	*Unsup*		
Assamese	*sMBR*	-	63.9	62.2
Assamese	*sMBR*	*Lattice Entropy*	63.4	61.6
Bengali	*sMBR*	-	66.3	64.1
Bengali	*sMBR*	*Lattice Entropy*	65.8	63.8
Zulu	*sMBR*	-	65.9	67.3
Zulu	*sMBR*	*Lattice Entropy*	65.7	67.2
Tamil	*sMBR*	-	76.3	74.8
Tamil	*sMBR*	*Lattice Entropy*	76.1	74.6

Summary

- Conditional entropy is shown to be a good sequence-discriminative criterion for semi-supervised training.
- Without explicit filtering of data, the method is shown to outperform self-training methods.
- Multilingual-inspired DNN architecture used for semi-supervised training. This works better than sharing single output softmax layer.

Future work

- While we can get small gains using lattice posterior confidences, we need to work on better confidence metrics to get larger improvements in semi-supervised training.
- Use sMBR-like criterion for training on unlabelled data.
- Study the effect of amount of unlablled data versus the labelled data and the effect of the quality of the unlabelled data.

References

G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean. In *ICASSP 2013*, pages 8619–8623, May 2013.

J. T. Huang. PhD thesis, 2012.

Z. Li and J. Eisner. In *Conf. on Empirical Methods in NLP*, volume 1, pages 40–51. Association for Computational Linguistics, 2009.

L. Mathias, G. Yegnanarayanan, and J. Fritsch. In *ICASSP (1)*, pages 109–112, 2005.

K. Vesely, M. Hannemann, and L. Burget. In *ASRU 2013*, pages 267–272. IEEE, 2013.

Thank you!