Main ingredients of the recipe to learn RNN to extract lists of characters (1D) from images (2D):

- Multi-Directional LSTM layers: 2D recurrence, 4 possible directions.
- Parameterized subsampling layers with filter sizes somehow adapted to the input resolution: 2D convolutions without overlapping.
- Collapse layer: from 2D to 1D flattened list of posterior log-probs.
- "CTC" training: Optimization of NLL by considering as equiprobable all possible alignments between the target characters and the outputs (including a not-a-character output).
- The framewise character probabilities estimated by the RNN (“optical model”) can be used along with a lexicon and/or a language model.

Curriculum Learning

**Key idea**: Carefully choosing training samples so as to start simple and gradually increase the complexity.

**Goal**: Converge faster, and hopefully generalize better in the end.

- Same idea in animal training (in shaping), cognitive science.

- The main question is: How to estimate if a training samples is a priori easy?

**Proposal**

Probability to draw a training sample $\{\text{image } X_i, \text{ target sequence } Y_i\}$:

$$P_i (\text{train on } (X_i, Y_i)) = \frac{1}{N_i \cdot \text{shortness}(Y_i)^\lambda}$$

where

- $N_i = \sum \text{shortness}(Y_i)$: normalization constant
- $\text{shortness}(Y_i) = \frac{1}{\max(m_i | Y_i |)} \in [0, 1]$ (if $\lambda = 1$, avoid focusing on extremely short sequences)
- $\lambda \geq 0$: hyper-parameter to tune how much short lines are favoured. Start with $\lambda = 3$ and progressively move towards uniform distribution ($\lambda = 0$) after the first 5 epochs.

Curriculum Learning for Handwritten Text Line Recognition

**Handwritten Text Recognition: Motivation for Curriculum Learning**

Applying RNN on lines is practical to recognize handwritten text: Unlike smaller units (words, characters), lines can be robustly extracted with automatic segmentation.

Three stages when learning a RNN directly from lines of text:

- **Slow beginning**: spend time without recognizing any character.
- **Steep acceleration**: efficient learning.
- **Remaining problem**: how to speed up training in the last stage? (learning plateau in the end)

**Experiments**

**Databases**

<table>
<thead>
<tr>
<th>Database</th>
<th>Language</th>
<th>X-axis unit</th>
<th># of target characters</th>
<th># of labelled words</th>
<th># of labelled images</th>
<th>Feature dimension (Input)</th>
<th>Feature dimension (Output)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>Arabic</td>
<td>10</td>
<td>5,567</td>
<td>200,000</td>
<td>10</td>
<td>100x100</td>
<td>100x100</td>
</tr>
<tr>
<td>OpenHaRT</td>
<td>Arabic</td>
<td>10</td>
<td>5,567</td>
<td>200,000</td>
<td>10</td>
<td>100x100</td>
<td>100x100</td>
</tr>
</tbody>
</table>

**Performance assessment**

- Convergence curves (valid set): $X$-axis unit: # of target characters browsed
- Optimized cost: Negative Log-Likelihood (NLL)
  $$\text{NLL} = \sum_{i=0}^{N-1} \text{NLL}(Y_i | X_i)$$
  $$\text{CER} = \frac{\sum_{i=0}^{N-1} \text{CER}(Y_i | X_i)}{N}$$

**Results (test set)**

<table>
<thead>
<tr>
<th>Database</th>
<th>Language</th>
<th>X-axis unit</th>
<th># of target characters</th>
<th># of labelled words</th>
<th>Feature dimension (Input)</th>
<th>Feature dimension (Output)</th>
<th>Performance assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM</td>
<td>Arabic</td>
<td>10</td>
<td>5,567</td>
<td>200,000</td>
<td>100x100</td>
<td>100x100</td>
<td>0.020 0.084 0.5000</td>
</tr>
<tr>
<td>OpenHaRT</td>
<td>Arabic</td>
<td>10</td>
<td>5,567</td>
<td>200,000</td>
<td>100x100</td>
<td>100x100</td>
<td>0.274 1.361 0.4917</td>
</tr>
</tbody>
</table>

**Figure**: English Handwritten Text Recognition (IAM)

**Figure**: French Handwritten Text Recognition (Rimes)

**Figure**: Arabic Handwritten Text Recognition (OpenHaRT)

**Conclusions**

- Curriculum Learning can halve training times of RNN for text recognition of lines. By removing the plateau in the beginning, it also makes training supervision easier.
- Remaining problem: how to speed up training in the last stage? (learning plateau in the end)