Connectionist Temp
Labelling Unsegment
Recurrent Neu

Thomas Mesnard

Probabilisitc Graphical Mod

Abstract

Many real-world sequence learning tasks require the prediction of sequences of labels from noisy, unsegmented input data. Recurrent neu-

Main Idea

RNNs are powerful learners for sequences, but:

- Standard methods need pre-segmented training data

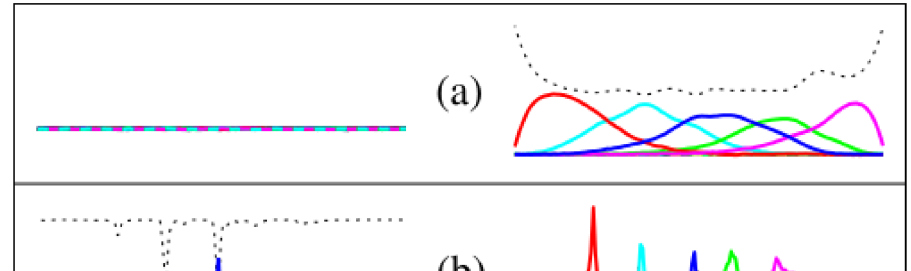
nporal Classification: nted Sequences with eural Networks

ard, Alex Auvolat

lodels Project, MVA Master

Recurrence equations

We define the following notation:
y_k^t: output at time t for symbol k
l: label, l': label with blanks
Initialization:



mented input data. Recurrent neural networks (RNNs) are powerful sequence learners that would seem well suited to such tasks. However, because they require pre-segmented training data, and post-processing to transform their outputs into label sequences, they cannot be applied directly. CTC is a method for training RNNs to label unsegmented sequences directly, thereby solving both problems.

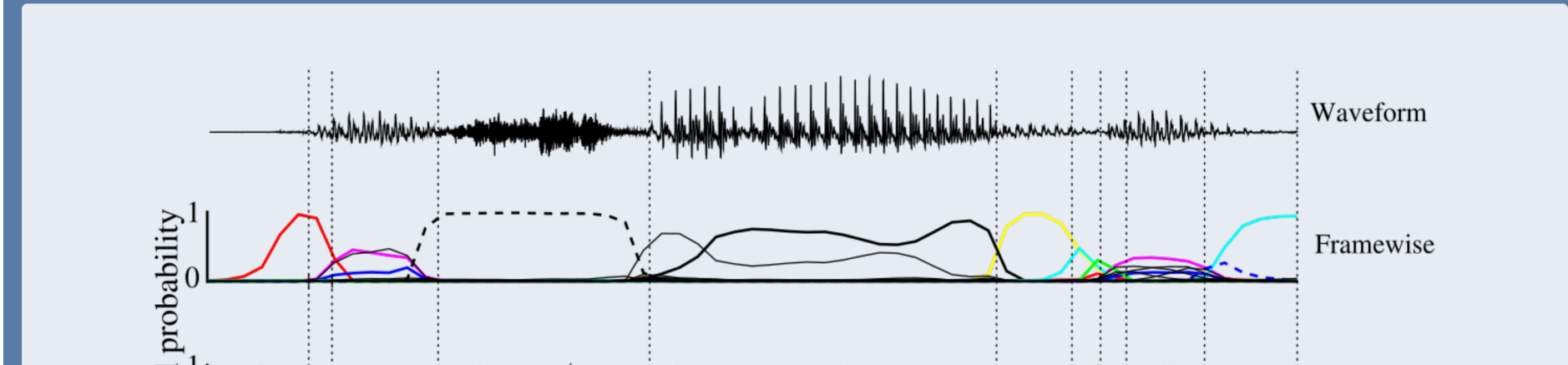
- Standard methods need pre-segmented training data
- Need for complex post-preprocessing

CTC solves this problem:

- Able to train RNNs using unsegmented training data
- Learns the segmentation automatically
- Provides directly usable output

This method is now extremely used, even by Google!

The problem and how CTC solves it



Initialization:

$$\alpha_1(1) = y_b^1$$
$$\alpha_1(2) = y_{l_1}^1$$
$$\alpha_1(s) = 0, \forall s > 2$$

Recurrence relation:

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y_{l'_s}^t & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y_{l'_s}^t & \\ \text{otherwise} & \end{cases}$$

$$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

Finally, we have:

$$p(l|x) = \alpha_T(|l'|) + \alpha_T(|l'|-1)$$

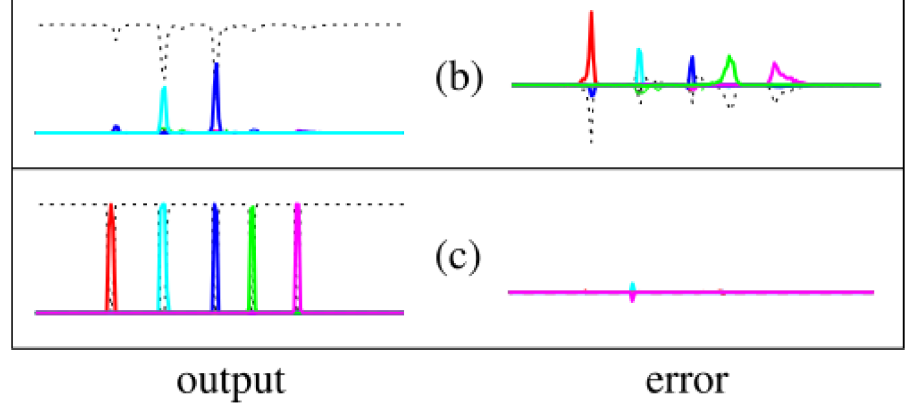


Figure 4: Evolution of the CTC error signal

To avoid numerical underflow, at each step t:

$$C_t = \sum_s \alpha_t(s) \quad \hat{\alpha}_t(s) = \frac{\alpha_t(s)}{C_t}$$

Other solution: do calculations in the logarithmic domain.

Toy dataset

We first tried our implementation on a simple task:

$$1^*2^*3^*4^*5^* \rightarrow 1$$
$$1^*2^*3^*2^*1^* \rightarrow 2$$

TIMIT

We then tried on the classical TIMIT dataset:

- Raw speech signal dataset

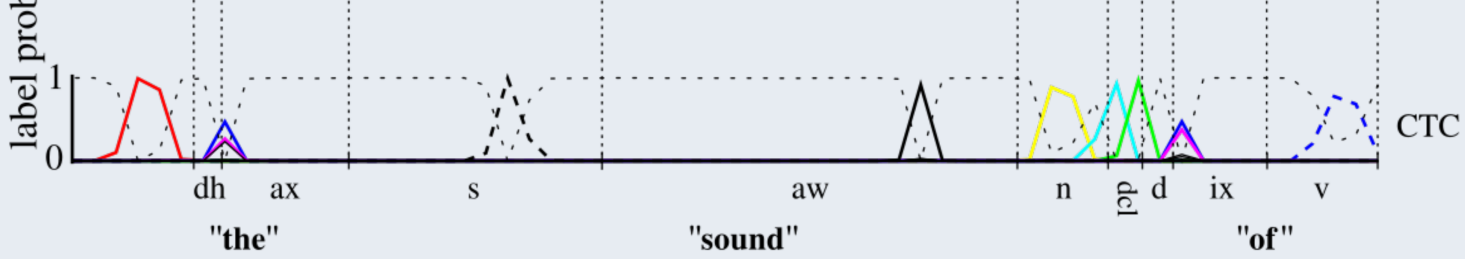


Figure 1: Output of classic framewise phoneme classification and RNN trained with CTC

Model

- Cost function for RNNs
- RNN outputs probabilities for the different symbols, plus blank symbol
- Many possible alignments for the correct label (shorter than input)
- Dynamic programming: sums all the possible alignments
- Provides gradients for the RNN to learn a good alignment

CTC is a dynamic programming algorithm that calculates the following sum:

$$\alpha_t(s) = \sum_{\substack{\pi \in N^T: \\ \mathcal{B}(\pi_{1:t}) = l_{1:s}}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}}$$

Where $\mathcal{B}$ is the transform that removes blanks and duplicates.

$1^*2^*3^*4^*5^* \rightarrow 1$

$1^*2^*3^*2^*1^* \rightarrow 2$

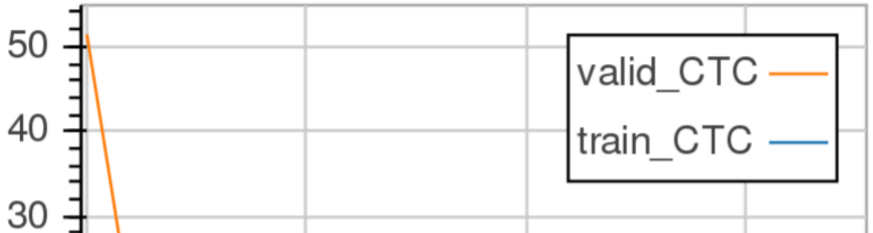
$5^*4^*3^*2^*1^* \rightarrow 3$

$5^*4^*3^*4^*5^* \rightarrow 4$

- A RNN can easily solve this
- It needs to read the full sequence before predicting a label
- CTC provides satisfactory results

Results	**train**	**valid**
Sequence length	5 – 20	5 – 20
Error rate	0.62	0.63
Mean edit distance	1.0	1.1
Errors per character	0.08	0.09

Table 1: Performances of CTC on our toy dataset



- Raw speech signal dataset
- Labelled by phonemes or by words
- 4120 sentences
- Average audio length: 50000 samples
- Avg. sentence length: 38 phonemes

Model:

- Convolution layers on raw signal
- Bidirectional LSTM layers
- Dropout and noise for regularization
- CTC cost function

This model avoids hand-crafted feature extraction on the speech signal. However it is extremely complicated to train such models. Our model hasn't converged yet.

Contact Information

learn a good alignment

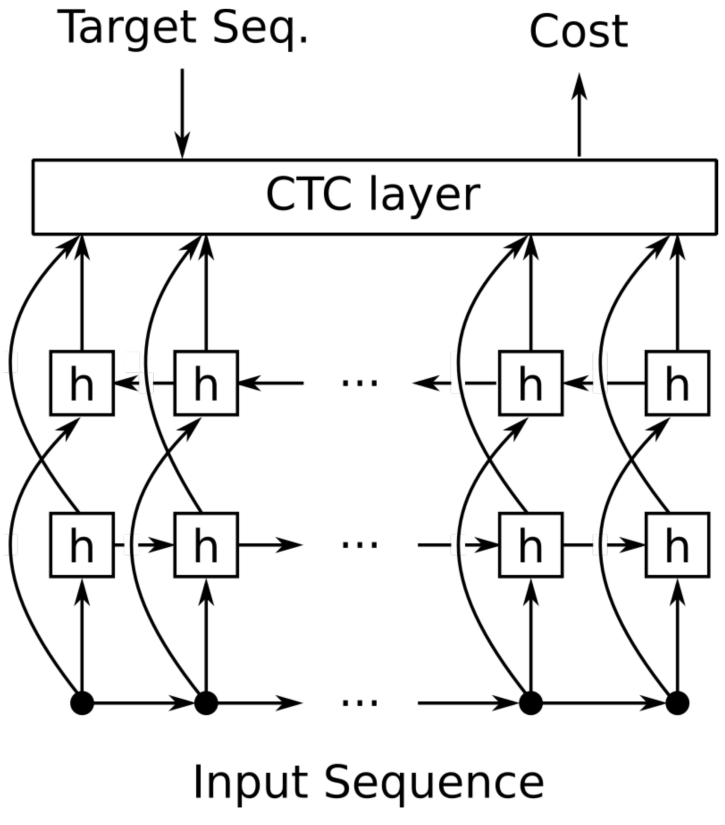


Figure 2: Simple bidirectional RNN model with CTC cost layer

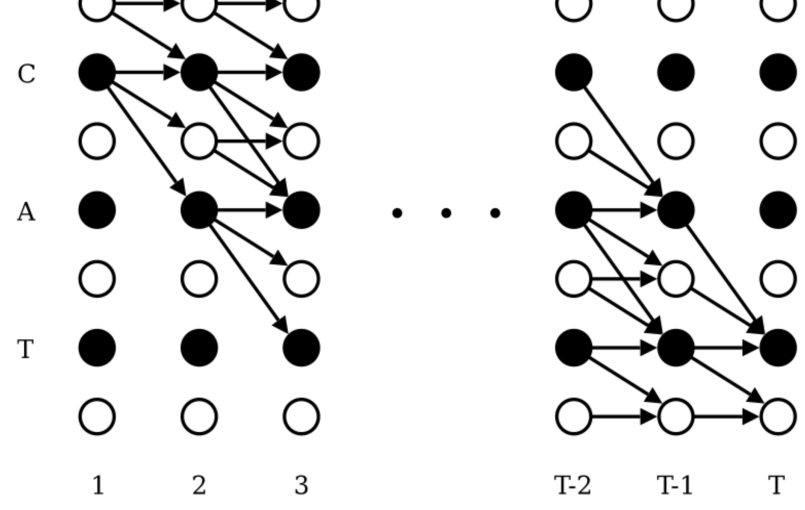


Figure 3: Computation graph for $\alpha_t(s)$ (corresponds to an unrolled automaton)

Tools used for our implementation:

- Theano (GPU computation library)
- Blocks (deep learning framework)

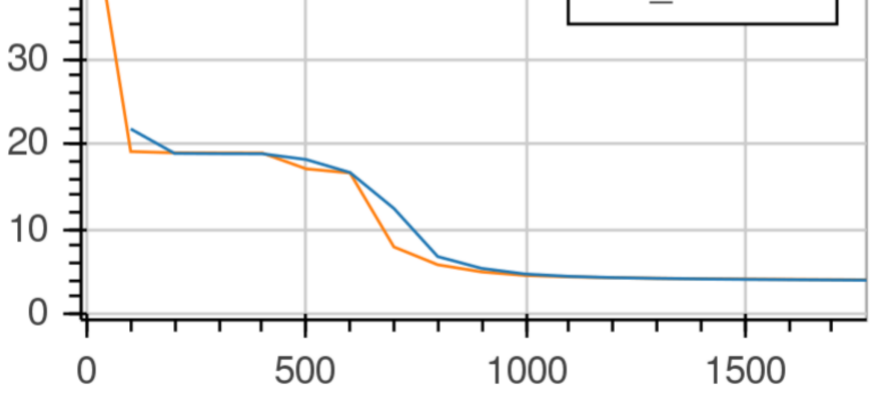


Figure 5: Training and validation cost of the CTC model (negative log likelihood)

Conclusion

CTC is a very powerfull model, and also has a nice mathematical formulation. It is also very used in practice (most successfull applications: speech recognition, handwriting recognition).

- Web: `http://github.com/thomasmesnard/CTC-LSTM`
- Email: `thomas.mesnard@ens.fr` `alex.auvolat@ens.fr`

References

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.