Journal of Computer and Knowledge Engineering, VOL. 1, NO.2, 2017.
DOI: 10.22067/cke.v1i2.69769

Modeling Intra-label Dynamics and Analyzing the Role of Blank in Connectionist Temporal Classification

Ashkan Sadeghi Lotfabadi

Kamaledin Ghiasi-Shirazi Ahad Harati*

Abstract: The goal of many tasks in the realm of sequence processing is to map a sequence of input data to a sequence of output labels. Long short-term memory (LSTM), a type of recurrent neural network (RNN), equipped with connectionist temporal classification (CTC) has been proved to be one of the most suitable tools for such tasks. With the aid of CTC, the existence of per-frame labeled sequences are no longer necessary and it suffices to only knowing the sequence of labels. However, in CTC, only a single state is assigned to each label and consequently, LSTM would not learn the intra-label relationships. In this paper, we propose to remedy this weakness by increasing the number of states assigned to each label and actively modeling such intra-label transitions. On the other hand, the output of a CTC network usually corresponds to the set of all possible labels along with a blank. One of the uses of blank is in the recognition of multiple consecutive identical labels. Assigning more than one state to each label, we can also decode consecutive identical labels without resorting to the blank. We investigated the effect of increasing the number of sub-labels with/without blank on the recognition rate of the system. We performed experiments on two printed and handwritten Arabic datasets. Our experiments showed that while on simple tasks a model without blank may converge faster, on real-world complex datasets use of blank significantly improves the results.

Keywords: Connectionist Temporal Classification; Handwriting Recognition; Recurrent Neural Networks; Multidimensional Long Short Term Memory; Blank.

1. Introduction

Labeling unsegmented sequences is one of the most significant and common problems in the field of artificial intelligence. Handwriting, speech and gesture recognition are examples of this problem. Some solutions to this problem have been given by probabilistic graphical models like HMM (Hidden Markov Model). However, HMMs are generative models whereas labeling a sequence is a dicriminative task.

On the other hand, RNNs (Recurrent Neural Networks) can be trained discriminatively and have a strong structure which learns the data and time dependencies. However, RNNs need pre-segmented data for training. A traditional solution has been to combine HMMs with RNNs. As we mentioned above, HMMs are generative models and they are not the best choice for a discriminative task like sequence labeling. Another solution is CTC which is a newer framework than HMMs. We can consider the output of RNN as a probability distribution on all the possible label sequences and then we get an objective function to maximize the true labeling probability [1].

One of the main issues confronted by RNNs is the vanishing/exploding gradient problem[2,3]. LSTM (Long Short Term Memory) [4,5] was designed to solve this problem. Moreover, BLSTM (Bidirectional LSTM) [6,7] and MDLSTM (Multidimensional LSTM) [8] are two other generalizations of LSTM and are proposed to learn bidirectional and multidimensional contexts, respectively. Since LSTM is a kind of RNN, it is possible to combine CTC with LSTM [1, 8-15].

Woellmer [10,16] proposed to combine DBNs (Dynamic Bayesian Network) and CTC to learn more complex relations like finding keywords in speech or text. In addition, there are some generalizations for CTC like ECTC (Extended CTC) [15] which consider a consistency to evaluate frame-to-frame visual similarities in CTC. Another generalization is HCTC (Hierarchical CTC) [17] which is composed of several layers of CTC in which each layer has a special task to learn a specific context of a sequence. In addition to RNNs, CTC can be combined with graphical models like LDCRF (Latent-Dynamic Conditional Random Field) [18].

One problem with CTC is that it leaves the task of learning the dynamics within each label to the underneath RNN. In this paper (This paper is the extension of our previous paper [19] A. S. Lotfabadi, K. Ghiasi-Shirazi, and A. Harati, "Modeling intra-label dynamics in connectionist temporal classification," in 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE), 2017, pp. 367-371.), we propose to model each label as a sequence of hidden internal labels and show how these internal labels can be learned. We postulate that by splitting each label to several hidden sub-labels, the task of the underneath RNN will become simpler and the overall accuracy increases. The output of the underneath network consists of all possible labels plus blank. The blank plays two roles in the network. Firstly, it allows recognition of consecutive identical labels and secondly, it frees the network from predicting the label of a sub-sequence until it has gathered enough evidence. Nevertheless, we will show that by considering several states for each label, the network can recognize contingent identical labels without using blank.

The organization of the paper is as follows: Section 0 introduces CTC, its mathematical formulation and its training algorithm. In Section 0, we introduce the Multi-state CTC

Manuscript received: October 25, 2017. Accepted December 22, 2017.
A. Sadeghi Lotfabadi, Department of Computer Engineering Ferdowsi University of Mashhad, Iran, a.sadeghilotfabadi@mail.um.ac.ir.
K. Ghiasi-Shirazi, epartment of Computer Engineering Ferdowsi University of Mashhad, Iran, k.ghiasi@um.ac.ir.
*A. Harati, Department of Computer Engineering Ferdowsi University of Mashhad, Iran. k.ghiasi@um.ac.ir.

(M-CTC) in which we propose splitting each label into multiple states/sub-labels. In section IV, we investigate the role of blank in standard CTC and the proposed M-CTC. In Section V we report our experiments on M-CTC with/without blank. We conclude the paper in Section VI.

2. Connectionist Temporal Classification (CTC)

CTC was proposed in 2006 by A. Graves [1] to train RNNs on unsegmented sequences. Prior to CTC, training RNNs on sequential data required the label to be specified for every frame of the input sequence. CTC revolutionized this process by making training possible when only a label sequence was given for the whole input sequence, without knowing the alignment between input and label sequences [20].

Assume that the number of outputs in the underneath RNN for each frame is equal to the number of labels plus one (for blank, or no label). A softmax layer normalizes these outputs before the last hidden layer sends them to CTC:

$$y_k^t = \frac{e^{a_k^t}}{\sum_{k'} e^{a_{k'}^t}} \tag{1}$$

where a_k^t is the kth output of the RNN in frame t and y_k^t is the normalized output. To obtain the probability of path (π), we use equation (2):

$$p(\pi \mid \boldsymbol{x}) = \prod_{t=1}^{T} p(\pi_t \mid \boldsymbol{x}) = \prod_{t=1}^{T} y_{\pi_t}^t \tag{2}$$

Paths are mapped to a label sequence by function F. This function removes the same consecutive labels and blank. For example, $F(a-ab-) = F(-aa--abb) = aab$ in which- means blank. Therefore, the probability of a special labeling like l is the sum of the probabilities of all paths which are mapped by F to l as shown in (3):

$$p(\boldsymbol{l} \mid \boldsymbol{x}) = \sum_{\pi \in F^{-1}(\boldsymbol{l})} p(\pi \mid \boldsymbol{x}) \tag{3}$$

The number of related paths to a specific labeling grows exponentially with respect to the length of the input sequence. However, it can be solved by dynamic programming and the algorithm is similar to the forward-backward in HMM [1]. Fig. 1 illustrates the CTC structure for all paths which map to the labeling ‘SUN’.

Loss function L(s) in CTC is the negative log probability of correctly labeling of all the training examples.

$$L(S) = -\ln \prod_{(\boldsymbol{x},\boldsymbol{z}) \in S} p(\boldsymbol{z} \mid \boldsymbol{x}) = -\sum_{(\boldsymbol{x},\boldsymbol{z}) \in S} \ln p(\boldsymbol{z} \mid \boldsymbol{x}) \tag{4}$$

The derivation of the loss function with respect to the networks parameters is done by using the backpropagation through time algorithm. So, it is possible to train the network by any gradient-based nonlinear optimization algorithm [20].

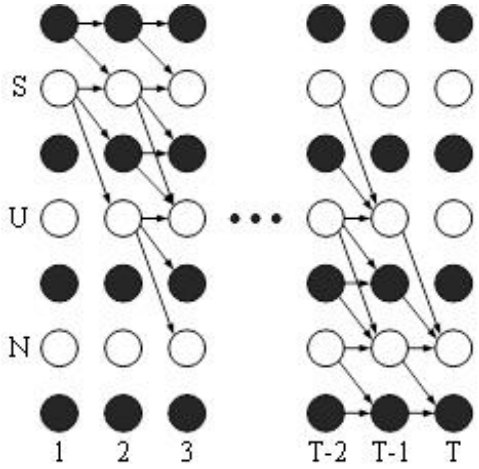

Fig. 1. CTC structure for ‘SUN’. Black and white units show *blank* and labels, respectively. The allowed paths which lead to ‘SUN’ labeling are determined by arrows.

3. Multi-state CTC (M-CTC)

Learning temporal context plays an important role in sequence labeling. So, the network should be able to maximally learn relations between frames. As we mentioned earlier, RNNs have the vanishing/explosion gradient problem. Therefore, we use LSTM which solves this problem. In most sequences the relations between frames are bidirectional. It means that future information is as important as past information. Thus, it is better to use networks which use both past and future information. Accordingly, we choose MDLSTM [8] which is a generalization of LSTM since MDLSTM can learn long-range dependencies in all spatio-temporal dimensions.

As we noted before, CTC was proposed to help RNNs (which is MDLSTM in our case) in training with unsegmented data. Considering Fig. 1, it is supposed that the MDLSTM determines the probability of each label at time t and the labels are given in the exact order from the left side which in this example are English alphabets. CTC considers 1 state for each label. So, the error is calculated by the CTC for each label at time t. Therefore, MDLSTM just learns the extrinsic dynamics between the labels. However, if we consider n state for each label in the CTC, the network not only learns extrinsic dynamics between the labels, but also learns intrinsic dynamics of labels and the relations of internal components of each label (which from now on we refer to as sub-labels). This increase in the number of states leads to better and more detailed learning of MDLSTM. An example of considering 2 states for each label class is illustrated in Fig. 2.

However, an equal number of sub-labels for each label may not be the best choice. It is better to determine the number of sub-labels according to the data length of that label the class. It means that a long data set needs more sub-labels than a short data set. Fig. 3 provides more details about this idea which illustrates 3 handwriting words from the IFN/ENIT [21] dataset. In this example, the number of sub-labels is determined based on average label length. By the above explanation, the number of sub-labels for letters ‘ث’, ‘ر’, ‘ا’ and ‘م’ are 5, 4, 2 and 5, respectively.

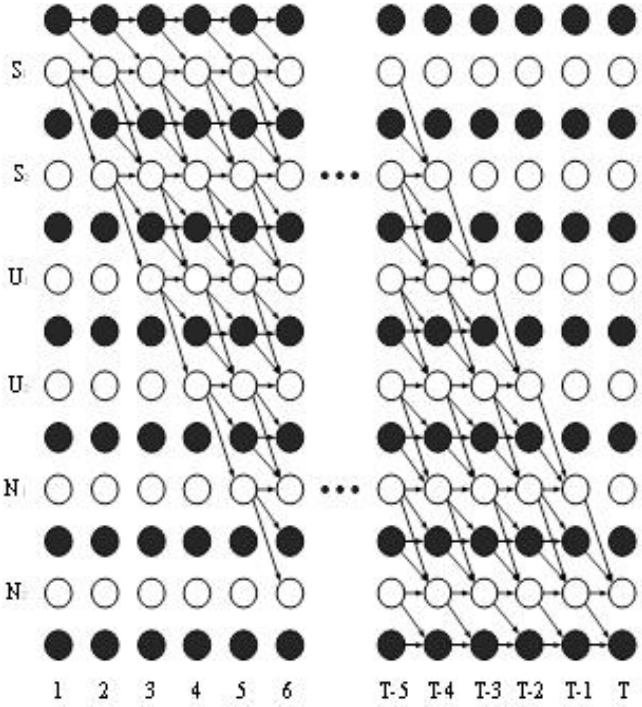

Fig. 2. CTC structure after considering 2 state for each label class

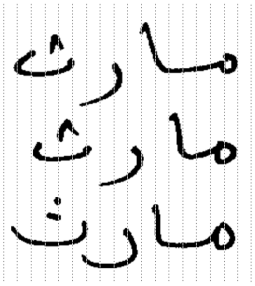

Fig. 3. Determining number of sub-labels according to label length. The gaps between vertical lines correspond to the frames.

4. Role of Blank

The output of the network beneath CTC consists of all possible labels plus the blank. Blank plays two roles in the network. Firstly, it prevents deletion of identical consecutive labels. For example, without blank CTC would recognize the word 'accuracy' as 'acuracy', recognizing the two consecutive letters 'c' as one. Secondly, blank frees the network from the oblige of predicting a label at each frame. For example, in the task of speech recognition, silences and short pauses between the utterance of phonemes can be recognized as blank (see page 64 of [20]). In the following, we will investigate the role of the blank when one models each label by multiple states in CTC.

According to Eq. (3), the probability of a labeling for the whole sequence is obtained by summing up the probabilities of all paths leading to that labeling. A path is an assignment of labels (possibly blank) to each frame. We define the function F from paths to sequences of labels as a mapping that removes blanks and repetitive labels. For example, paths 'oo-f-fff', '--off-f---', and '---o-ff--fffff-' are mapped to the sequence label 'off'. Please note that in these examples, frames which are labeled 'f' form two groups which are separated by one or multiple blanks. Having only a single group of labels 'f', the path would have been mapped to the label sequence 'of'. In fact, one of the reasons for provisioning the blank in CTC was to avoid misrecognition of words with consecutive repetitive words. Fig. 4 shows the structure of a CTC for recognizing the word 'off'. As it can be seen, there is no link between the two instances of label 'f' and all paths should pass through the intermediate blank label.

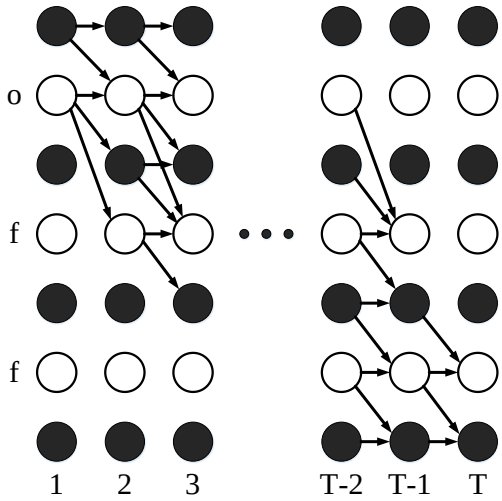

Fig. 4. CTC structure for 'off'. There is no link between the two instances of label 'f' and all paths should pass through the intermediate blank label.

In Fig. 4, which exemplifies the structure of standard CTC, each label is modeled by exactly one state (shown as nodes in the Figure). In this paper, we propose to model each label with several states/sub-labels. Having multiple states for each label automatically guarantees that consecutive labels would be correctly recognized, eliminating one of the reasons behind considering the special label blank. Assume that we have modeled each label with two states and have eliminated blank from CTC, arriving at the CTC structure is shown in Fig. 5. Now, to map a sequence of frames to a label, it is necessary to see both of its sub-labels in order, which guarantees correct recognition of consecutive repetitive labels. For example, paths '$o_1o_1o_2f_1f_2f_2f_2f_1f_2$' and '$o_1o_1o_2f_1f_1f_2f_1f_2f_2$' both yield the sequence label '$o_1o_2f_1f_2f_1f_2$'. Modeling each label with multiple sub-labels eliminated the first reason for the existence of the blank. But, what about the second reason which was to free the network from predicting a (non-blank) label at each frame? In the following, we perform experiments with/without blank to answer this question.

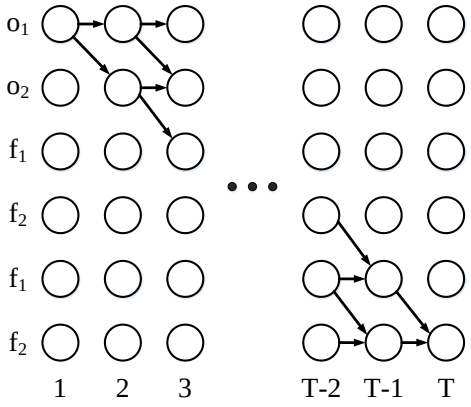

Fig. 5. CTC structure for 'off' by considering 2 state for each label and removing blank.

5. Experiments and Results

In this section, we explain the experiments used to evaluate the performance of the proposed method. The goal of these experiments is to investigate the role of the number of sub-labels and inclusion/exclusion of blank in the recognition accuracy. To evaluate the standard CTC, we use RNNLIB [22] which has been developed by inventors of CTC themselves. The neural network at the bottom of CTC consists of three hidden layers each one having four parallel LSTM layers. Each LSTM layer consists of several LSTM cells, the number of which is identified by "Hidden Size" in the tables of our experiments. The input data of each hidden layer is obtained by moving a window on the output neurons of the preceding layer in four directions. The sizes of these windows are designated by "Input Block" and "Hidden Block" in input and hidden layers, respectively. We also use some subsampling hidden layers whose sizes are identified by "Subsampling Size". In all cases, we use the gradient descent training algorithm.

5-1. Printed Digits Dataset

To evaluate the proposed method in the task of discovering the true sub-labels of each label, we generated an artificial dataset in which each label actually encompasses two sub-labels. In this dataset, we have 10 different labels corresponding to digit sequences '01', '03', '11', '12', '21', '31', '33', '41', '43', and '52', each label being made from two digits. The dataset contains 1000 samples of sequences of 3 labels (6 digits). Fig. 6 shows a sample image from this dataset. We use 800 samples for training, 100 for validation, and the remaining 100 for testing.

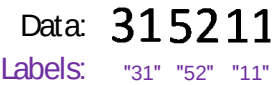

Fig. 6. An example of printed digits dataset with true labels.

We performed 7 different experiments on this dataset. The parameters of four of these experiments which were performed with blank are shown in Table 1 and are titled from A to D. There are three other experiments which are identical to the experiments B, C, and D, except that now the blank is removed. The reason that experiment A has no counterpart with blank removed is that in experiment A the number of sub-labels is equal to one and, according to the explanations of Section 4, the blank is essential to the recognition of consecutive repetitive labels. The number of states for each label in the experiments B, C, and D was chosen as 2, 4, and 10 respectively. The other fact important to note is the difference between the values of "Input Block" and "Hidden Block" in experiment D. The reason behind this difference is that since the number of sub-labels is very high, the length of the input data to CTC should be long enough to visit all the sub-labels. Therefore, the length of the input data to CTC in experiment D should be slightly more than that of the other three experiments. In fact, we believe that one of the reasons that in our experiments M-CTC has been shown to be superior to CTC, in contrast to the experiments reported in [23], is the adjustment of these parameters.

Table 1. Parameters For Printed Digits Experiments

Experiment	A	B	C	D
Hidden Size	2, 10, 50	2, 10, 50	2, 10, 50	2, 10, 50
Subsample Size	6, 20	6, 20	6, 20	6, 20
Hidden Block	3x4, 2x4	3x4, 2x4	3x4, 2x4	2x4, 2x4
Input Block	3x4	3x4	3x4	2x4
Learn Rate	1e-4	1e-4	1e-4	1e-4
momentum	0.9	0.9	0.9	0.9
optimizer	Steepest ascent	Steepest ascent	Steepest ascent	Steepest ascent
Number of Sub-labels	1	2	4	10

Because of the simplicity of this dataset, in all experiments, the accuracy of 100% was obtained. For this reason, we compare method based on their speed at reaching a solution. Explicitly, we compare the error of different methods at the end of a certain epoch. Table 2 illustrates the results of our experiments. This Table has been filled based on the error obtained at epoch 25. As stated previously, experiment A has no "without-blank" counterpart. The experiments show that increasing the number of sub-labels has decreased the error, obtaining the best results in the experiment D. The other important observation is the effect of removing blank. The results show that the accuracy of experiments without blank is superior to those with a blank. This shows that in this simple example (in which data for each label are artificially generated by concatenating data of two sub-labels), not only the blank label does not improve the results, but it also has increased the time of getting error zero. An example of the output of a model with 4 sub-labels and without blank is shown in Fig. 7. In this Figure, two consecutive instances of label '52' are recognized without any problem.

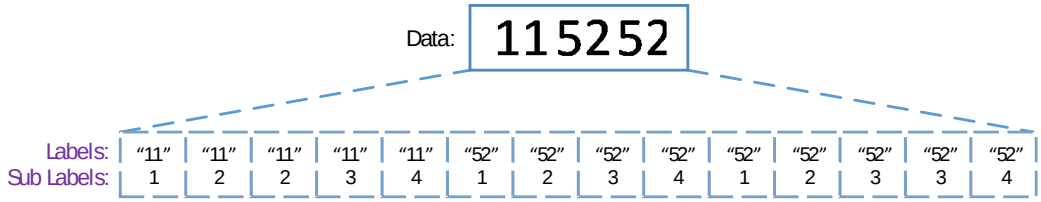

Fig. 7. MDLSTM output for a data from printed digits. In this model blank is removed and each label has 4 sub-labels. Each square shows a frame and includes the labels and sub-labels.

Table 2. Results For Printed Digits

Experiment	With blank						Without blank					
	CTC Error			Label Error %			CTC Error			Label Error %		
	Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test
A	7.52	7.60	7.67	87.63	88.94	87.90	-	-	-	-	-	-
B	12.15	11.75	11.79	74.57	73.72	72.65	0.29	0.30	0.31	1.79	2.12	1.68
C	1.00	0.84	0.82	0.28	0.22	0.13	0.33	0.36	0.35	0.06	0.05	0.05
D	0.19	0.19	0.19	0.00	0.00	0.00	0.21	0.21	0.20	0.00	0.00	0.00

5-2. IFN/ENIT Dataset

This dataset consists of 32249 samples from the handwritten images of 937 towns in Tunisia. Each sample in this dataset has a label sequence, each label being chosen from 120 possible choices of letters of alphabet, digits, and punctuation signs. This dataset has five segments named 'a' to 'e'. We perform our experiments on segment 'a'. We first randomly select a subset of segment 'a' consisting of 1000 samples and perform our experiments on this subset. Then we repeat the experiments on the whole segment 'a'. A sample of data of this segment is shown in Fig. 5.

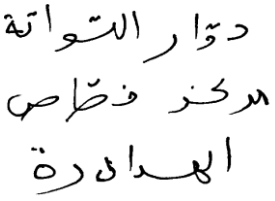

Fig. 8. Some sample handwritten words from the INF/ENIT dataset.

We perform four experiments on the 1000-sample data and the whole samples of segment 'a'. Details of the three experiments A, B, and C (in which the blank label is present) are shown in Table 3. The first three experiments are named A, B, and C and the fourth experiment is similar to B with the difference that the blank label is removed. The reason that experiments A and C are not repeated without blank is that in these experiments some of the labels have only one state in CTC. In experiment C, the number of sub-labels ranges between 1 and 3 in proportion to the average length of data of that label. The values of "Input Block" and "Hidden Block" differ between the three experiments and is chosen in a way that ensures that there are enough data for all sub-labels of the input data. This dataset is much harder than the printed digits dataset and so the number of cells in LSTM, i.e. the value of "Hidden Size", is chosen in a way that gives the best results for each experiment. Since by increasing the number of sub-labels the network should learn more details about data, the number of LSTM cells should increase accordingly. In all experiments, we use the "early stopping" method to avoid overfitting. If the error rate on validation set does not decrease for 40 consecutive epochs, we stop training and return the network with the lowest validation error.

Table 3. Parameters For IFN/ENIT Experiments

Experiment	A	B	C
Hidden Size	2, 10, 50	4, 20, 80	4, 15, 64
Subsample Size	6, 20	6, 20	6, 20
Hidden Block	3x4, 2x4	2x4, 2x4	2x3, 1x3
Input Block	3x4	2x4	3x3
Learn Rate	1e-4	1e-4	1e-4
momentum	0.9	0.9	0.9
optimizer	Steepest Descent	Steepest Descent	Steepest Descent
Number of Sub labels	1	2	1 or 2 or 3

5-3. 1000 sample subset of IFN/ENIT Dataset

The 1000 samples of this dataset have been chosen randomly from segment 'a' of the IFN/ENIT dataset. We use 800 samples for training, 100 for validation, and 100 for testing. The results of the experiments on this dataset are shown in Table 4. By comparing experiments A and B one can observe that increasing the number of sub-labels from one to two has increased recognition accuracy. In addition, by comparing experiments B and C one can see that the appropriate choice of the number of sub-labels has improved the results. Now,

we investigate the role of the blank label by considering experiment B with and without blank. It can be seen that the results obtained with blank are much higher than those obtained without it. Another important observation is that without blank, the training error is much higher (in addition to the test error). This shows that removal of blank leads to a learning machine with much lower capacity (possibly due to optimization issues). Therefore, we designed another experiment in which we removed the stopping condition and allowed the network to obtain a much lower error on the training set. However, the results obtained by this method did not differ considerably from those obtained by early stopping.

5-4. Segment 'a' of the IFN/ENIT Dataset

Segment 'a' of the IFN/ENIT dataset consists of 6537 samples from which 5702 are used for training, 426 for validation, and 409 for testing. The results are given in Table 5. By comparing this Table with Table 4, we see that by increasing the number of training samples the model is much better optimized and much higher accuracy results have been obtained over the validation and test sets. In addition, similar to the previous experiments, we have obtained better results in experiment B in comparison with experiment A, because of using two sub-labels for each label. We have achieved the best results in experiment C in which the number of sub-labels is chosen in proportion to the average length of each label. By comparing experiment B in the two cases with and without blank we see that the results with blank are much better than those without blank. This shows that the use of blank in datasets with complex data leads to improved recognition accuracy.

5-5. Analyzing the effect of blank

In the previous sections, we reported the results of our experiments on the printed digits and IFN/ENIT datasets. The first dataset was very simple as every sample contained three labels with equal lengths and widths. We observed that the use of blank deteriorated the speed of obtaining a solution. In contrast, in experiments that did not use blank, much better results had been obtained in early epochs. Because of the simplicity of this dataset, all experiments ended with 100% accuracy. However, the IFN/ENIT dataset was much more complex: having labels with different average lengths, and 12 times more labels. For example, Fig. 3 shows different styles of letters which differ in length and writing style. Because of this complexity, the network should have a flexible structure which can learn all states of each label. The results showed that the use of blank has a huge effect on improving the recognition accuracy. It can be deduced that even when the number of states is more than one, use of blank improved the recognition results on real-world datasets. After doing this work, we found that similar observations have been done in [23], confirming the superiority of RNN-CTC models using blank over those that do not. In contrast to the results reported in [23], we found that the combination of increasing the number of states per label and using blank gives the best results. This difference in observations may be due to the fact that we have modified the structure of the beneath neural network appropriately to cope with the increase in the number of states in M-CTC.

Table 4. Results For 1000 Data From IFN/ENIT

Experiment	With blank						Without blank					
	CTC Error			Label Error %			CTC Error			Label Error %		
	Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test
A	0.53	17.45	11.93	0.22	40.22	41.35	-	-	-	-	-	-
B	0.57	28.27	19.56	0.03	33.77	33.30	76.69	95.60	83.88	68.71	76.26	81.75
C	0.96	25.89	18.85	0.51	31.64	30.25	-	-	-	-	-	-

Table 5. Results For set 'a' of IFN/ENIT

Experiment	With blank						Without blank					
	CTC Error			Label Error %			CTC Error			Label Error %		
	Train	Val	Test	Train	Val	Test	Train	Val	Test	Train	Val	Test
A	0.41	6.69	5.46	0.49	19.45	17.98	-	-	-	-	-	-
B	0.33	9.43	8.18	0.07	13.28	12.85	25.33	36.99	37.73	29.13	46.24	46.28
C	0.32	8.36	7.53	0.15	12.66	10.86	-	-	-	-	-	-

5-6. Conclusions and future works

In this paper, we extended CTC to model and learn intra-label relations. We achieved this goal by increasing the number of states in CTC for each label. In other words, we considered several states/sub-labels for each label. Experimental results showed that the proposed extension improves the recognition accuracy. In addition, we studied the role of blank. We showed that by increasing the number of states, models without blank can learn repetitive consecutive labels. Our experiments showed that while on simple tasks a model without blank may converge faster, on real-world complex datasets use of blank significantly improves the results. Although we restrict the model to see the sub-labels in true order during training, there is not a similar obligation during test time. In other words, it is possible during test time that the model predict the second sub-label without seeing the first sub-label. In future, we want to analyze the impact of this discrepancy at training and testing time on the recognition accuracy.

References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks", *in Proceedings of the 23rd international conference on Machine learning*, pp. 369-376, 2006.

[2] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult", *IEEE transactions on neural networks*, vol. 5, pp. 157-166, 1994.

[3] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies", *ed: A field guide to dynamical recurrent neural networks. IEEE Press*, 2001.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, pp. 1735-1780, 1997.

[5] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks", *Journal of machine learning research*, vol. 3, pp. 115-143, 2002.

[6] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition", *in International Conference on Artificial Neural Networks*, pp. 799-804, 2005.

[7] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures", *Neural Networks*, vol. 18, pp. 602-610, 2005.

[8] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks", *in Advances in neural information processing systems*, pp. 545-552, 2009.

[9] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition", *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, pp. 855-868, 2009.

[10] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Spoken term detection with connectionist temporal classification: a novel hybrid ctc-dbn decoder", *in 2010 IEEE International Conference on Acoustics*, Speech and Signal Processing, pp. 5274-5277, 2010.

[11] M. Wöllmer, B. Schuller, and G. Rigoll, "Probabilistic ASR feature extraction applying context-sensitive connectionist temporal classification networks", *in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7125-7129, 2013.

[12] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks", *in 2013 IEEE international conference on acoustics*, speech and signal processing, pp. 6645-6649, 2013.

[13] M. Wöllmer, F. Weninger, J. Geiger, B. Schuller, and G. Rigoll, "Noise robust ASR in reverberated multisource environments applying convolutive NMF and Long Short-Term Memory", *Computer Speech & Language*, vol. 27, pp. 780-797, 2013.

[14] A. Graves and N. Jaitly, "Towards End-To-End Speech Recognition with Recurrent Neural Networks", *in ICML*, pp. 1764-1772, 2014.

[15] D.-A. Huang, L. Fei-Fei, and J. C. Niebles, "Connectionist Temporal Modeling for Weakly Supervised Action Labeling", arXiv preprint arXiv:1607.08584, 2016.

[16] M. Woellmer, B. Schuller, and G. Rigoll, "Keyword spotting exploiting long short-term memory", *Speech Communication*, vol. 55, pp. 252-265, 2013.

[17] S. Fernández, A. Graves, and J. Schmidhuber, "Sequence Labelling in Structured Domains with Hierarchical Recurrent Neural Networks", *in IJCAI*, pp. 774-779, 2007.

[18] A. A. Atashin, K. Ghiasi-Shirazi, and A. Harati, "Training LDCRF model on unsegmented sequences using Connectionist Temporal Classification", *arXiv preprint arXiv:1606.08051*, 2016.

[19] A. S. Lotfabadi, K. Ghiasi-Shirazi, and A. Harati, "Modeling intra-label dynamics in connectionist temporal classification", *in 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE)*, pp. 367-371, 2017.

[20] A. Graves, "Neural Networks," in Supervised Sequence Labelling with Recurrent Neural Networks, ed: Springer, pp. 15-35. , 2012

[21] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri, "IFN/ENIT-database of handwritten

Arabic words", *in Proc. of CIFED*, pp. 127-136, 2002.

[22] A. Graves, "RNNLIB: A recurrent neural network library for sequence learning problems", [OL][2015–07-10], 2013.

[23] T. Bluche, H. Ney, J. Louradour, and C. Kermorvant, "Framewise and CTC training of Neural Networks for handwriting recognition", *in Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pp. 81-85. , 2015.