

Handwriting Recognition with Connectionist Temporal Classification

Chibuzor John Amadi

¹ University of Pavia, Pavia

² University of Milano-Bicocca, Milan

³ University of Milano-Statale, Milan

Abstract. Sequence learning tasks where the input data is unsegmented can be seen in various real-world activities. Among these activities are speech recognition and handwriting text recognition. Recurrent Neural Networks (RNNs) are very useful sequence learners that could be used to carry out this tasks; however, RNNs require pre-segmented training data. This paper explores a method for training an RNN model to label unsegmented sequences directly. This model, the Connectionist Temporal Classification (CTC) model, through a detailed analysis of CTC's operational mechanics, with its unique use of a blank label and the forward-backward algorithm, evaluates its performance on handwriting recognition.

Keywords: Un-segmented sequences · forward-backward algorithm · CTC

1 Introduction

Unsegmented sequence data is a important problem to be invested in as it arises in many common real-world cognitive activities such as speech recognition, gesture recognition, and handwriting recognition, which we are investigating in this paper. To label unsegmented sequences, we must draw inspiration from models that have been the predominant blueprint for sequence labelling. Models like the Hidden Markov Model (HMM) and Conditional Random Fields are graphical models that have shown success in sequence labelling. Despite the drawbacks of HMM, recent developments have discovered a way of merging HMMs with recurrent neural networks (RNNs) in a "hybrid approach" [1]. This hybrid approach used HMMs to model long-range sequential structure and the neural net to provide localised classification. This system, however, inherited the drawbacks of the HMM.

The paper will explore a method for labelling sequence data with RNNs eliminates the need for pre-segmented data and post-processed outputs. Essentially, we aim to interpret the network outputs as a probability distribution over all possible label sequences, conditioned on a given input sequence. We will refer to the task of labelling unsegmented data and our use of RNNs as connectionist temporal classification (CTC) [2]

2 Model Architecture

2.1 CNN + RNN

The U-Net [1] The model of this paper is a deep architecture that leverages convolutional neural networks (CNNs) and recurrent neural networks (RNNs). This hybrid model begins with feature extraction by the CNN. The feature extractor is made up of 9 residual blocks. Residual blocks introduced by Kaiming He et al. in their influential paper on ResNet [3](Residual Networks), provide a solution to this problem through the introduction of shortcut connections that perform identity mapping. These shortcut connections facilitate the flow of gradients during training, making it easier to train deeper networks without suffering from the vanishing gradient problem. The residual block uses the relu activation function and computes;

$$y = \text{relu}(F2(\text{relu}(F1(x))) + x) \quad (1)$$

Where $F(1,2)$ are the composition of a convolution and a batch normalization. The skip connection $\dots + x$ is replaced by a 1×1 convolution when the shape of the two addends do not match. The blocks include an optional stride, and a final dropout. The initial layers of the residual block expand the channel to capture low-level features. Subsequent blocks maintain and increase the depth of the network, improving its ability to learn complex features. Designing the feature extractor takes into consideration that it reflects a focus on capturing and emphasizing textual features in images. By preserving horizontal resolution longer than vertical, the network is optimized to handle the linear nature of text, making it adept at recognizing sequences of characters and spaces as they appear in typical handwritten lines.

The second half of this hybrid model is the recurrent neural network, specifically a bi-directional Long Short Term Memory Model (LSTM). The LSTM comes after feature extraction and its job is to analyse the features in a sequence, it captures the context and dynamics of the sequence, which is crucial for accurate character recognition. The LSTM being bidirectional helps the model understand fully the content of the sequence in both direction, backwards and forward. This provides deeper insight into the data by considering both past and future states simultaneously.

The output of the LSTM is passed to a linear layer, which is responsible for classifying each time step into a character class, including a special class for blanks in CTC-based models. Before training the model, some addition functions are implemented. These functions include; a function to remove blanks and repetitions, a function to calculate the accuracy of the prediction.

3 Evaluation

3.1 Experiment details

For the purpose of this project, we will use a processed version of the "IAM handwriting database" [4]. The dataset contains 5000 lines of handwritten text.

4500 for training and 500 for test. For each of the 5000 lines, there is a text file, which the correct text, that indents accurately with the handwritten text. Important to note that although majority of the text lines are sentences, some are simply English letters, numbers or may contain symbols. This symbols will be taken into consideration as well.

For training this model, the Adam [5] optimizer was utilized and the loss function will be the CTC loss function. The accuracy and the edit distance are the performance metrics to be used on the model. The CTC loss function can be mathematically described as seen in figure 2. The CTC loss aims to maximize the probability of the correct label sequence L given the predicted sequence Y . The loss is defined as the negative log probability of L given Y :

$$CTCLoss = -\log P(L : Y) \quad (2)$$

To calculate the probability of the L given Y we must compute the alignments of the blanks, all possible alignments and the forward-backward algorithm. as seen figure 3.

However, using log-softmax[6] with CTC changes the computational approach from direct probability manipulations to log-probability manipulations. It simplifies the implementation by improving numerical stability and is typically the recommended approach in practice for implementing CTC loss. A possible drawback to the CTC loss on the log-softmax is that the accuracy might suffer but not too detrimentally.

The edit distance[7] also called the Levenshtein distance as the performance metric computes the minimal number of editing operations i.e, insertions, deletions and substitutions to transform the prediction in the target sequence. The edit distance can be effectively and efficiently implemented with a dynamic algorithm.

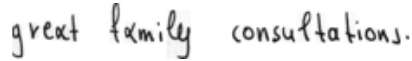


Fig. 1. An example of one line of the Handwritten text, This will have corresponding text "great family consultations"

$$CTC Loss = -\log \left(\sum_{\pi \in \Pi} \prod_{t=1}^T y_t^{\pi_t} \right)$$

Fig. 2. where N represents all possible valid paths that can be mapped to L , and y_t is the probability assigned by the network at time t to the character at position n_t in path n .

4 Results Analysis and Conclusion

The result of the model compared the actual prediction, the CTC decoding process, and the output of the CTC decoder. Investigating these results, some observations could be made; First, the handwritten lines with very few words and a presence of a symbol were highly misclassified, words like "Dr. " were classified as "O.-". Second, looking at that image, words written in cursive were also highly misclassified, as CNN found it difficult to extract single characters on it. The final observation was that although long sentences had few letter misidentifications, they performed on average better with the actual prediction than scanty sentences or single words or letters.

Experimenting with the LSTM. A unidirectional LSTM was used in place of the bidirectional LSTM. This change produced a major difference in the loss of the model. The loss on the CTC LSTM bidirectional on 100 epochs was $0.03 + 0.2879 - 0.03$ while on the CTC LSTM unidirectional on 100 epochs it was $0.03 + 1.0034 - 0.03$.

The accuracy on the overall model are as follows: 81% on the training set and 69% on the test set. This model performs poorly with respect to generalization. Addition parameters are necessary to handle the models overfitting. Introducing an additional dropout layer in the residual block, with similar dropout rate (0.2) and an L2 regularizer in the training process were the initial thoughts to improve generalization. The accuracy of the model after including the tools to improve generalization are as follows: 61% accuracy on the train set with a loss of 0.33 and 65% on the test set. The results were realized after training the model for a hundred epochs. The drop from 81% to 61% can be attributed to the fact that the dropout and L2 regularizer introduces noise and constraints to the training process and makes it slightly more difficult for the model to fit data perfectly. However, the drop in the training accuracy as a drawback also showed an advantage as it achieved its goal with the model performing better on the test set 65%. With this, we have seen that the regularization techniques are very useful in this task as it helps in the generalization of the model.

To conclude, I believe with more computational resources available at this instance, the model presented in this paper can perform even better. Data augmentation, deeper models and more training of the model and its hyper-parameters can lead to an optimal accuracy and minimization of the loss.

```

1) who is sitting, .....
2) person, with a trigger .....
3) the social life .....
4) written in book .....
5) it seems silent that, though .....
6) psychologist I am constantly .....
7) the definition of .....
8) hear the end of .....
9) Sir A. Hall can hear, .....
10) for my country .....

```

Fig. 3. Showing the Original text of the sentence with the CTC decoding process

