Impact of Training LSTM-RNN with Fuzzy Ground Truth

Martin Jenckel[1,2], Sourabh Sarvotham Parkala[2], Syed Saqib Bukhari[1] and Andreas Dengel[1,2]
[1]*German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany*
[2]*TU Kaiserslautern, Kaiserslautern, Germany*

Keywords: Document Analysis, OCR, LSTM, Fuzzy Ground Truth.

Abstract: Most machine learning algorithms follow the supervised learning approach and therefore require annotated training data. The large amount of training data required to train state of the art deep neural networks changed the methods of acquiring the required annotations. User annotations or completely synthetic annotations are becoming more and more prevalent replacing careful manual annotations by experts. In the field of OCR recent work has shown that synthetic ground truth acquired through clustering with minimal manual annotation yields good results when combined with bidirectional LSTM-RNN. Similarly we propose a change to standard LSTM training to handle imperfect manual annotation. When annotating historical documents or low quality scans deciding on the correct annotation is difficult especially for non-experts. Providing all possible annotations in such cases, instead of just one, is what we call fuzzy ground truth. Finally we show that training an LSTM-RNN on fuzzy ground truth achieves a similar performance.

1 INTRODUCTION

With the rise of neural networks and deep learning, collecting large amounts of training data and their annotation became one of the major challenges in machine learning. Instead of handcrafting annotations for data sets with a couple thousand data points, often the data is collected from social media sources like Twitter, Flickr or YouTube (Wang et al., 2011; Althoff et al., 2013; You et al., 2015) and instead of handcrafted annotations user generated tags, keywords or comments are used. This allows for data sets with millions of data points with comparably low effort. While this works great for images and videos, it is rather difficult to employ it for other data types like historical documents.

For images and videos most people can describe what they see fairly well, but for historical documents it can be difficult to read characters like the examples shown in Figure 1 and provide accurate annotations. Even more so if the language on the document is old, not used anymore or changed significantly compared to its modern form. Additionally historical documents often show various degradations making it even harder to identify the characters without the right expertise. In practice this makes annotating historical documents time consuming and therefore also expensive with a single page often costing 100USD or more to transcribe.

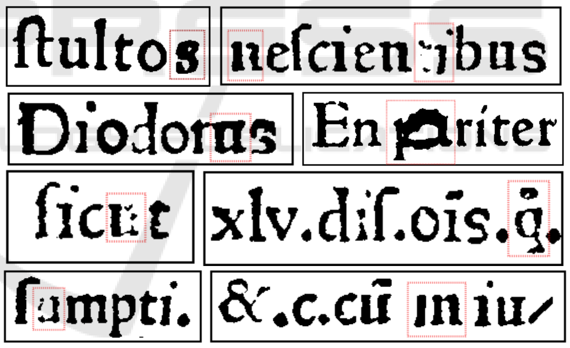
Figure 1: Examples for historical documents with degraded or otherwise hard to identify characters.

In this paper we propose a new way of annotating historical documents for the purpose of LSTM-RNN based Optical Character Recognition (OCR) systems. Annotating ambiguous characters, either through high levels of degradations or ambiguity in the characters themselves, can often be done by identifying the correct word, surrounding words or the overall context of the sentence and text. Without this linguistic expertise it becomes difficult to make the correct decision. We propose, that instead of removing useful information from the annotation by possibly giving the wrong annotation, the transcription should contain all possibilities. LSTM-RNN based OCR systems can

Jenckel, M., Parkala, S., Bukhari, S. and Dengel, A.
Impact of Training LSTM-RNN with Fuzzy Ground Truth.
DOI: 10.5220/0006592703880393
In *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2018)*, pages 388-393
ISBN: 978-989-758-276-9

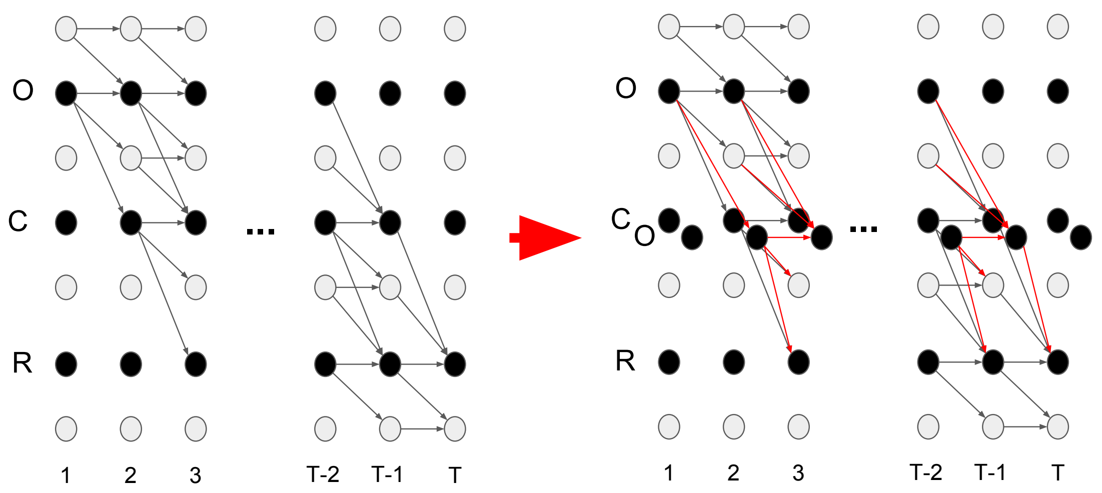

Figure 2: In combination with LSTM-RNN the CTC loss function considers all possible alignments between networks outputs and ground truth. Compared to the classic case (left), fuzzy ground truth (right) effectively increases the number of possible alignments (red).

use ”fuzzy ground truth” without a significant loss in overall accuracy. This allows for transcribing historical documents without extensive knowledge about the language and requires only basic knowledge about the character shapes. At the same time it reduces the overall annotation time because ambiguous characters do not have to be resolved by using additional tools and time.
In Section 2 the details about how LSTM-RNNs can handle the fuzzy annotation compared to normal annotation is explained. In Section 3 there is a description of the experimental setup and an introduction to the data set. Lastly in Section 5 we will conclude the performance and give an outlook on further research topics.

2 Long-Short-Term Memory - Recurrent Neural Networks

A Long-Short-Term-Memory - Recurrent Neural Network (LSTM-RNN) is a RNN consisting of LSTM-cells. They were introduced to solve the problem of vanishing gradients during gradient descent training of classical RNN networks (Werbos, 1990; Hochreiter and Schmidhuber, 1997). LSTM-RNNs excel at sequence learning tasks and can align unsegmented data with their corresponding ground truth when combined with Connectionist Temporal Classification (CTC) loss (Graves et al., 2006). This properties make them especially useful for OCR where they produce state of the art results (Ul-Hasan et al., ; Karayil et al., 2015; Simistira et al., 2015).

An LSTM-cell has three gating mechanisms that regulate the impact of the input via the input gate, the previous cell state via the forget gate and the output via the output gate. Especially the forget gate and the cells internal state allow it to ”remember” information through multiple time steps and similarly ”remember” errors backwards through time during gradient descent optimization.
The state equations for each LSTM-cell are as follows:

$$\begin{aligned}
f_t &= \sigma(w_{xf} \cdot x_t + w_{hf} \cdot h_{t-1} + b_f) && (1)\\
i_t &= \sigma(w_{xi} \cdot x_t + w_{hi} \cdot h_{t-1} + b_i) && (2)\\
v_t &= tanh(w_{xv} \cdot x_t + w_{hv} \cdot h_{t-1} + b_v) && (3)\\
o_t &= \sigma(w_{xo} \cdot x_t + w_{ho} \cdot h_{t-1} + b_o) && (4)\\
C_t &= f_t \cdot C_{t-1} + i_t \cdot v_t && (5)\\
y_t &= o_t \cdot tanh(C_t) && (6)
\end{aligned}$$

where

- f, i and o are the forget, input and output gate,
- C is the cell state and y the cell output, while t iterates of all time steps
- $b_j, j \in \{f, i, o, v\}$ are the bias units for the forget, input and output gate and the input squashing,
- w_{ij} are the weight connection between i and j,
- σ is the logistic sigmoid function given as: $\sigma = \frac{1}{1+exp(-x)}$
- and $tanh$ is the tangent hyperbolic function $tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$,

Storing information through multiple time steps allows the LSTM to learn temporal relations in the data.

In terms of OCR this means it inherently builds a statistical language model based on temporal relations of characters and words.
The most common architecture for LSTM-RNNs is bidirectional LSTM (BILSTM). Two LSTM-RNNs consisting of an input layer and a hidden layer go through the sequence from both sides simultaneously and feed into a single Softmax layer. Since the number of LSTM-cells in the input layer has to be fixed, all input sequences need the same input size at every time step.

2.1 Connectionist Temporal Classification Loss

The Connection Temporal Classification (CTC) loss allows RNNs to be trained on unsegmented data (Graves et al., 2006). Similar to other loss functions the idea is to maximize the likelihood, that the output of the network aligns with the given ground truth. For this all possible alignments between network output and ground truth have to be considered. This is done by aligning each ground truth character with a sequence of network outputs, while keeping the temporal order consistent. This can be understood as extending the ground truth to the same length of the input sequence by adding "blank" characters that refer to no output, or duplicating the same character multiple times which is equivalent to multiple consecutive inputs referring to the same character in the transcription. For example "....aa.b." and ".aabb...", with the "blank" label ".", are two possible extensions of the transcription "ab" to a sequence length of 8. The likelihood of a labeling l is therefore the sum over the probabilities of all possible extensions $T(l)$ to the length of the network output, given the network output x.

$$p(l|x) = \sum_{\pi \in T(l)} p(\pi|x) \tag{7}$$

By using a forward-backward algorithm one can then calculate $p(l|x)$ as:

$$p(l|x) = \sum_{n} \alpha_t(l_n)\beta_t(l_n) \tag{8}$$

with $\alpha_t(l_n)$ and $\beta_t(l_n)$ being the forward and backward variables for the n-th symbol l_n in the labeling l at network output t. The LSTM-RNN can then be trained by maximizing the log-likelihood over all possible extensions. A more detailed mathematical discourse can be found in (Graves et al., 2006) and (Graves, 2012).

2.2 Training with Fuzzy Ground Truth

We propose using fuzzy ground truth for ambiguous characters rather than absolute ground truth. This can be especially useful when people without language expertise transcribe historical documents with degraded or otherwise ambiguous characters. Fuzzy ground truth means providing the CTC loss function with multiple options l_{n_1} and l_{n_2} for a specific element of the transcription l_n. Training on wrong annotation has two negative effects. It does not reinforce the correct label and reinforces the wrong label instead. This means network weights would be updated to produce the wrong output. By providing fuzzy ground we can prevent these wrong weight updates. The LSTM-RNN will not be able to learn which of the options is the correct label from the fuzzy annotation alone, since both will be correct in terms of CTC-loss. The intrinsic statistical language modeling of LSTM-RNNs however will generate a bias towards the more likely outcome.
For the CTC-loss multiple ground truth options represent additional possible extensions of the labeling (Figure 2). The forward and backward variables are defined as:

$$\alpha_t(l_n) = y^t_{l_n} \begin{cases} \sum\limits_{i=n-1}^{s} \alpha_{t-1}(i) & l_n \in \{b, l_{n-2}\} \\ \sum\limits_{i=n-2}^{s} \alpha_{t-1}(i) & \text{else} \end{cases} \tag{9}$$

$$\beta_t(l_n) = \begin{cases} \sum\limits_{i=n}^{n+1} \beta_{t+1}(i) y^t_{l_i} & l_n \in \{b, l_{n+2}\} \\ \sum\limits_{i=n}^{n+2} \beta_{t+1}(i) y^t_{l_i} & \text{else} \end{cases} \tag{10}$$

From equation 9 it becomes clear, that the forward variables $\alpha_t(l_{n_1})$ and $\alpha_t(l_{n_1})$ only differ in the network output $y^t_{l_{n_i}}$. The following forward variables $\alpha_{t_+}(l_n)$ therefore get the same input from $\alpha_t(l_{n_1})$ and $\alpha_t(l_{n_2})$ up to the factor $y^t_{l_{n_i}}$. This means we can instead write

$$\alpha_t(l_{n_{1/2}}) = y^t_{l_{n_{1/2}}} \begin{cases} \sum\limits_{i=n-1}^{s} \alpha_{t-1}(i) & l_n \in \{b, l_{n-2}\} \\ \sum\limits_{i=n-2}^{s} \alpha_{t-1}(i) & \text{else} \end{cases} \tag{11}$$

with $y^t_{l_{n_{1/2}}} = y^t_{l_{n_1}} + y^t_{l_{n_2}}$.
$\beta_t(l_{n_1})$ and $\beta_t(l_{n_2})$ do not depend on $y^t_{l_{n_i}}$ and are therefore the same. For any $\beta_{t_-}(l_n)$ we can therefore also replace every $y^t_{l_i}$ with the sum $(y^t_{l_{i_1}} + y^t_{l_{i_2}})$.

Table 1: Common confusions for historical documents in Latin script. All confusions are both ways.

p	*l*	*l*	*l*	*l*	*l*	*l*	*t*	*a*
P	*t*	1	*r*	*i*	ſ	/	*r*	ā
n	*n*	1	*c*	*a*	*a*	*b*	*f*	
m	*u*	*i*	*e*	*o*	*u*	*h*	ſ	

3 EXPERIMENTAL SETUP

Due to the high cost of manual transcriptions, rather than reannotating our data, we decided to synthetically generate fuzzy ground truth in two different scenarios based on existing annotations.

3.1 Scenario I

In the first scenario we consider annotations by an expert and compare it with synthetic non-expert and fuzzy ground truth annotations. In this experiment the expert's annotation has an accuracy of 100% and serves as a base line. The non-expert's annotation has a 50% chance of correctly choosing between two options for randomly selected characters. For the fuzzy ground truth both, the wrong and the correct transcription options are given. The fuzzy ground truth will therefore have exactly twice as many fuzzy characters as the non-expert's ground truth has errors in the worst case scenario. While the ambiguous characters are selected randomly, the possible confusions are selected to be realistic. For a full list of confusions see Table 1. All confusions are 5% and symmetric.

3.2 Scenario II

A second scenario for fuzzy ground truth arises from the work of (Jenckel et al., 2016). In their setup they combine clustering with BILSTM and manage to improve the ground truth generated via clustering through LSTM training.
K-Means clustering results in hard bounded clusters assigning each point to the closest cluster center. To generate fuzzy ground truth Gaussian mixture model (GMM) clustering is used instead. It is a probabilistic approach and well established within the machine learning community (Reynolds, 2015). GMM clustering assigns a probability $p(y = k_i|x)$ that a data point x is in cluster k_i. This will allow us to naturally generate fuzzy ground truth based on the posterior probabilities. The GMM algorithm optimizes the weighted sum of Gaussians $g(x|\mu, \sigma)$ with mean μ and covariance σ:

$$p(x|\lambda) = \sum_i w_i g(x|\mu_i, \sigma_i) \tag{12}$$

using an Expectation-Maximization (EM) algorithm. The algorithm is initialized with the results of a k-Means clustering.
Before clustering the text lines were segmented into individual characters using a connected-component based algorithm and then resized to uniform size of 32x32 pixel while keeping the aspect ratio constant. In both cases the algorithm follows the description given in (Jenckel et al., 2016). To reduce the high dimensionality of the raw features for the GMM a PCA has been added to the pipeline.

3.3 OCRopus

As a base for the LSTM implementation we used OCRopus[1], a complete OCR software package. Its implementation has shown to produce state of the art results while being easy to use (T. M. Breuel, A. Ul-Hasan, M. Al Azawi, F. Shafait, 2013). Beside the LSTM framework it provides us with a text line normalization method based on Gaussian filtering and affine transformation as described in (Yousefi et al., 2015).

3.4 Data

The data set consists of 100 binarized pages and the corresponding transcription of a Latin version of the 15th century novel "Narrenschiff"[2]. One of the main challenges when working with historical documents is the comparably small size of the data sets. In order to train on a maximum of training samples only 2 pages were used as the test data set leaving 3263 text lines with a total of 83780 characters for training. The test set consists of 103 lines with 2876 characters.

3.5 Parameters

In both scenarios some free parameters had to be set. For the clustering the number of components used in PCA as well as the number of cluster centers in the GMM had to be chosen. For the PCA the top 10 components were used. Setting the number of cluster centers for the GMM clustering to $k = 200$ follows the same strategy as described in (Jenckel et al., 2016). The idea is to largely overestimate the number of clusters during clustering. During the following annotation of the clusters, similar clusters provided with

[1]https://github.com/tmbdev/ocropy
[2]http://kallimachos.de/kallimachos/index.php

Table 2: Results for scenario 3.1. Comparison of the Character Error Rates (CER) for the LSTM-RNN training with perfect ground truth from an expert, worst case ground truth from a non-expert and fuzzy ground truth containing both transcription options.

	$T0$	$T1$	$T2$
$Expert$	0	2.9	2.9
$Non-Expert$	3.3	4.2	3.8
$Fuzzy$	6.7	2.7	3.0

the same label are indirectly merged.
For the LSTM implementation in OCRopus some parameters have to be set as well. OCRopus only supports a single hidden layer whose size is set to 100. Another parameter is the size of the input layer which is equivalent to the height of the text lines. As reported in (Jenckel et al., 2016), 48 is a reasonable value. Learning rate and momentum were kept at the default values of $1e-4$ and 0.9.

4 RESULT

For evaluation of the two proposed scenarios from section 3.1 and 3.2, we report the Character Error Rate (CER) of the model with the lowest CER on seen and unseen data. All CER are calculated using the Levenstein-distance. For seen data we use a 3 page subset of the training data called $T1$. The best model on $T1$ is then evaluated on our 2 page test set $T2$. The CER on the training data set $T0$ describes the error introduced to the ground truth by wrong annotations. The experts annotation serves as a base line and is considered to be perfect (0% error). For fuzzy ground truth instead we give the percentage of characters with multiple annotations.
The results for the first scenario are seen in Table 2. While the ground truth from an expert leads to state of the art CER for this data set, the non-experts ground truth with 3.3% total error leads to 4.2% CER after training and 3.8% on the test set. This is in line with the results from (Jenckel et al., 2016). Even though the error in the ground truth was increased by 3.3%, the resulting error compared to perfect ground truth only rises by about 1%. The LSTM trained with fuzzy ground truth however achieves almost the same accuracy as the 100% accurate ground truth from the expert, even though 6.5% of the data was fuzzy. The differences between the model trained with the experts transcription and the fuzzy transcription are within the normal variance.
In the second scenario we used the pipeline described in (Jenckel et al., 2016) but replaced the hard bounded k-Means clustering with the probabilistic GMM clustering. After segmentation, clustering and manual annotation our accuracy was rather low with a CER of 27.4%. For the fuzzy ground truth we assigned the label of every cluster with a posterior probabilities higher than 0.05. For comparison we also trained LSTM models on generated ground truth by using the label of the cluster with the highest probability to every character, the label of the second most probable cluster and the label of the third most probable cluster (if available). We evaluated on the same data sets $T1$ and $T2$. The results are shown in Table 3. Due to the high CER after clustering the CER are still high. However LSTM training increased the overall performance significantly, cutting CER by one third. Choosing the second most probable cluster is the better choice for some characters slightly improving the ground truth. However it also is worse for other data points and leads to a worse overall performance. Training on fuzzy ground truth performed on the same level as taking the most probable cluster for each data point, while outperforming the two alternative annotations. Similar to the previous scenario the rate of fuzzy characters in the ground truth was 6.5%.

Table 3: Results for scenario 3.2. Comparison of the Character Error Rates (CER) for LSTM-RNN training after GMM clustering. In the case of fuzzy ground truth the CER on $T0$ is the rate of characters with multiple non-identical transcriptions.

	$T0$	$T1$	$T2$
1*st*	32.9	24.5	22.1
2*nd*	32.4	26.6	27.3
3*rd*	33.5	27.7	28.9
$Fuzzy$	6.5	24.3	23.1

5 CONCLUSION AND OUTLOOK

In this paper we have proposed a time conserving change in annotating historical documents using fuzzy ground truth. We also showed that in theory that non-expert's annotations can lead to similar results like expert's ones. We also explained how LSTM-RNN with CTC can handle fuzzy ground truth and showed that training LSTM with fuzzy ground truth, where at least one of the transcription options is the correct ground truth, does not reduce the performance significantly. We conclude that the LSTM-RNN with CTC can successfully select the correct option from the provided possibilities through statistical language modeling.
With regards to the bad results when applying GMM clustering to the segmented data we conclude that GMM is not a good choice for this type of clustering.

However it provided us with a natural way of generating fuzzy ground truth in the context of the ”anyOCR” pipeline proposed in (Jenckel et al., 2016).
The scope of this paper only covered the feasibility of non-expert annotations which was shown through the use of synthetic data. Therefore we plan further evaluation with real non-expert annotations. While we have shown that training on fuzzy ground truth can be beneficial in the area of historical documents, further analysis on different scripts and document types is needed as well.
In the future we plan to further explore the possibilities of using fuzzy ground truth when training LSTM networks, like ground truth options with different segmentations like ”m” and ”in”. Another future focus will be on automatically generating fuzzy ground truth. While the proposed method reduces the need for language experts it is still costly to annotate the data by hand.

ACKNOWLEDGEMENTS

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project Kallimachos (01UG1415C).

REFERENCES

Althoff, T., Borth, D., Hees, J., and Dengel, A. (2013). Analysis and forecasting of trending topics in online media streams. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 907–916. ACM.

Graves, A. (2012). Supervised sequence labelling. In *Supervised Sequence Labelling with Recurrent Neural Networks*, pages 5–13. Springer.

Graves, A., Fernndez, S., Gomez, F. J., and Schmidhuber, J. (2006). Connectionist temporal classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *ICML'06*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. In *Neural Computation*, pages 1735–1780.

Jenckel, M., Bukhari, S. S., and Dengel, A. (2016). anyocr: A sequence learning based ocr system for unlabeled historical documents. In *23rd International Conference on Pattern Recognition (ICPR'16)*, Mexiko.

Karayil, T., Ul-Hasan, A., and Breuel, T. M. (2015). A Segmentation-Free Approach for Printed Devanagari Script Recognition. In *ICDAR*, Tunisia.

Reynolds, D. (2015). Gaussian mixture models. In *Encyclopedia of biometrics*, pages 827–832. Springer.

Simistira, F., Ul-Hasan, A., Papavassiliou, V., Gatos, B., Katsouros, V., and Liwicki, M. (2015). Recognition of Historical Greek Polytonic Scripts Using LSTM Networks. In *ICDAR*, Tunisia.

T. M. Breuel, A. Ul-Hasan, M. Al Azawi, F. Shafait (2013). High Performance OCR for Printed English and Fraktur using LSTM Networks. In *ICDAR*, Washington D.C. USA.

Ul-Hasan, A., Ahmed, S. B., Rashid, S. F., Shafait, F., and Breuel, T. M. Offline Printed Urdu Nastaleeq Script Recognition with Bidirectional LSTM Networks. In *ICDAR'13*, USA.

Wang, H., Klaser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. pages 3169–3176. IEEE.

Werbos, P. (1990). Backpropagation through time: what does it do and how to do it. In *Proceedings of IEEE*, volume 78.

You, Q., Luo, J., Jin, H., and Yang, J. (2015). Robust image sentiment analysis using progressively trained and domain transferred deep networks. In *CoRR*, volume abs/1509.06041.

Yousefi, M. R., Soheili, M. R., Breuel, T. M., and Stricker, D. (2015). A Comparison of 1D and 2D LSTM Architectures for Recognition of Handwritten Arabic. In *DRR-XXI*, USA.