Iterative Weighted Transductive Learning for Handwriting Recognition

George Retsinas[1], Giorgos Sfikas[2], and Christophoros Nikou[2] *

[1] School of Electrical and Computer Engineering
National Technical University of Athens, Greece
{gretsinas}@central.ntua.gr
[2] Department of Computer Science and Engineering
University of Ioannina, Greece
{sfikas,cnikou}@cse.uoi.gr

Abstract. The established paradigm in handwriting recognition techniques involves supervised learning, where training is performed over fully labelled (transcribed) data. In this paper, we propose a weak supervision technique that involves tuning the trained network to perform better on a specific test set, after having completed its training with the standard training data. The proposed technique is based on the notion of a reference model, to which we refer as the "Oracle", which is used for test data inference and retraining in an iterative manner. As test data that are erroneously labelled will be a hindrance to model retraining, we explore ways to properly weight Oracle labels. The proposed method is shown to improve model performance as much as 2% for Character Error Rate and 5% for Word Error Rate. Combined with a competitive convolutional-recurrent architecture, we achieve state-of-the-art recognition results in the IAM and RIMES datasets.

Keywords: Weak supervision, Handwriting Recognition, Transductive learning

1 Introduction

Today, the vast majority of state-of-the-art methods for document image processing tasks are learning-based methods [17]. This rule has very few exceptions [19], while the form of learning employed in most cases is supervised learning. Handwriting Text Recognition (HTR) adheres to this rule, and the most succesful HTR methods can be described as supervised deep learning methods. The immediate implication is that large labelled sets are required for succesful training. Labels in HTR are manually created transcriptions that can be accurate in terms of correspondence to the image data up to the level of either word, line

* This research has been partially co-financed by the EU and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call OPEN INNOVATION IN CULTURE, project *Bessarion* (T6YBΠ-00214).

or character, depending on the specifics of the method at hand. Creating large annotated datasets is understably time-consuming, and even more so if one takes also into account the labour required to manually record token pixel boundaries.

Using deep and supervised models for HTR comes thus with the shortcoming that large annotated datasets are required for model training. An issue of no lesser importance than this first issue, is that of model transferability. In particular, we cannot be a priori certain that our model trained on set A will generalize well enough to be practically applicable on set B. For example, set B may be written using a style that is not covered by the writing styles included in set A, or the topic of the text in set B may be such that a language model trained on set A does not apply to B well enough, and so on.

A multitude of techniques have been proposed to create more transferable models, either in or out of the context of document image processing [20, 25]. A strategy that can be helpful is to treat test data as if they were unlabelled training data, and proceed to employ techniques from the arsenal of semi-supervised learning. This strategy is referred to as transductive learning [28], and is closely related to self-training [4] and (standard) semi-supervised learning [21]. The main difference from the latter strategies is that the unlabelled data is not necessarily the test data. In all cases, care must be taken as using unlabelled data may just as well lead to a degradation of performance [2].

Concerning the application of transductive learning, let us note that the character of the task does play a role with respect to whether this type of learning can be effective. In HTR in practice, a test set is available for inference by the learning machine as a whole: for example, the test set can be made up of consecutive scanned lines of a given manuscript. This is not the case in other tasks, where the norm is to have test data appearing sequentially to the inference pipeline. Hence, HTR can be categorized as a task where we realistically have the option to treat our test set as a whole, instead of a disjoint set of objects. In the context of using transductive learning this is important, as it enables the option to adapt the initial model to the domain represented by the test set.

In this work, we propose and explore the applicability of a transductive learning technique, suitable for HTR. The proposed technique is based on multiple iterations of using the test data with updated labels. Also, we experiment on using a weight term on newly labelled text lines, in order to formulate a rule for including in the transduction retraining loss. We explore two different ways to perform weighting, and compare versus not using any weighting and including all new labels for retraining. We show that the proposed iterative, weighted transductive learning method can significantly improve results for a competitive CNN-RNN architecture. Our experiments show that the proposed model leads to state-of-the-art performance for HTR line recognition on the IAM and RIMES datasets.

This paper is structured as follows. In section 2, we review related work for transductive learning and HTR. In section 3.2, we present the proposed method and the base HTR architecture that we use for our experiments, presented and discussed in section 4. We close the paper with concluding remarks in section 5.

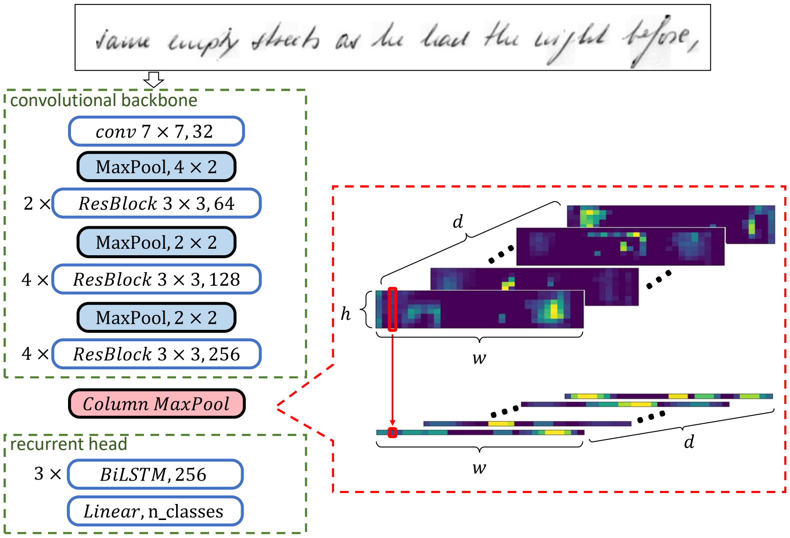


Fig. 1. Employed model architecture. We use a CNN-RNN architecture, with a column-wise max pooling layer forming the link between the convolutional and the recurrent-sequential component.

2 Related work

Techniques that are closely related to transductive learning come from the domain of semi-supervised learning. As a notable example, self-training has been previously in the context of semi-supervised learning and HTR [4]. The trained model is used to first infer labels on unlabelled data, of which the most confident labels are used for retraining the original model. Different retraining rules are tested, according to which part of the newly labelled words are picked to train the model. These rules use a different confidence threshold, ranging from a more conservative value to a rule that simply uses all words for retraining.

Transductive learning in HTR, in the sense of including a training subprocess that uses labelled *test* data (in constrast to classical semi-supervised learning, that assumes a separate subset that is used for that purpose) has been shown to be especially useful in cases where the size of the dataset considered is very limited [6]. Concerning character-level recognition, transductive learning has been shown to improved performance for the task of digit recognition on the well-known MNIST dataset [12]. A more recent work for Tibetan handwritten character recognition has validated the usefulness of transductive learning [9], by modifying a number of semi-supervised and domain adaptation algorithms in the task context. Along with domain adaptation-based approaches, the authors use self-supervision and a method based on adversarial training.

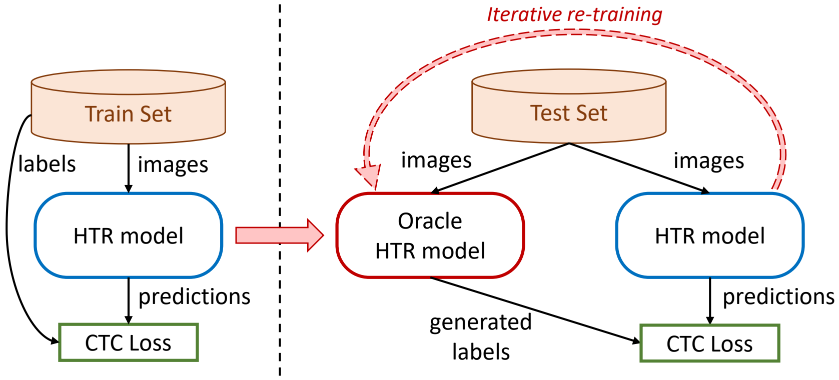


Fig. 2. Proposed learning pipeline. Left: A model suitable for HTR is trained on the training set (see fig. 1 for details about the model architecture). This model will subsequently be the "Oracle model". Right: The Oracle is used to produce labels for the test set. These labels are used as to train a copy of the Oracle. After training completes, the newly trained model replaces the Oracle, which in turn produces updated test set labels.

3 Proposed HTR model and transductive learning technique

In this section, we describe a transduction-based technique. We start by the assumption of a specific HTR architecture, the results of which will be shown to be further improved with the proposed technique.

3.1 Proposed model architecture

The model that we will use to test the proposed technique can be characterized as a CNN-RNN architecture (overview in fig. 1). CNN-RNN can be broadly defined as a convolutional backbone being followed by a recurrent head, typically connected to a CTC loss. CNN-RNN variants have given routinely very good results for HTR [3, 24].

In our model, the convolutional backbone is made up of standard convolutional layers and ResNet blocks, interspersed with max-pooling and dropout layers. In particular, we have a 7 × 7 convolution with 32 output channels, a series of 2 3 × 3 ResNet blocks with 64 output channels, 4 3 × 3 ResNet blocks with 128 output channels and 4 3 × 3 ResNet blocks with 256 output channels. The standard convolution and the ResNet blocks are all followed by ReLU activations, Batch Normalization 2 × 2 max pooling of stride 2 and dropout. Overall, the convolutional backbone accepts a line image and outputs a tensor of size $h \times w \times d$. This output is then squashed by column-wise max-pooling into a $w \times d$-sized tensor. With this operation, the feature map is ready to be processed by the recurrent component, while achieving model translation invariance in the vertical direction [24]. The recurrent component consists of 3 stacked Bidirectional Long Short-Term Memory (BiLSTM) units of hidden size 256. These are followed by a linear projection layer, which converts the sequence to a size equal to the number of possible tokens, $n_{classes}$. Finally, the softmax output which tops the recurrent head converts these feature sequence into a sequence of probability distributions. The softmax output, sized $w \times n_{classes}$, is fed into a Connectionist Temporal Classification (CTC) loss. Note that the CTC logic provides translation invariance in the horizontal sense, complementing the vertical invariance that is related to the pooling operation at the end of the convolutional backbone. Decoding of the output during evaluation is performed by a greedy selection of the characters with the highest probability at each step.

3.2 Transductive learning cycle

The motivation behind the proposed methodology is clear: adapt an already trained network to the test set characteristics (e.g. writing style) by iteratively re-training a new network with labels generated by the pre-trained network. Of course, we cannot be guaranteed that the generated labels are correct. Therefore, such an approach has an important prerequisite regarding the performance of the pre-trained network. Specifically, the network should be able to recognize an adequately large percentage of the existing characters. Having a model that provides more character hits than misses, i.e. a per-character percentage of over 50% accuracy, can potentially improve the overall performance. We consider that such a bound can be easily achieved by modern HTR systems.

Considering the task at hand, the proposed transductive learning can be described as follows. First off, we assume that a training set with labelled data (text lines) is available, along with a test set with unknown labels. The CNN-RNN model is first trained on the training data, whereupon after training is complete, we freeze this model's weights and from now this model is referred to as the "Oracle" model. We use the Oracle on the test set on inference mode, in order to produce labels for the test set. The test set adaptation is performed by training a new network with identical architecture to the Oracle. Note that the only prior information about the text manifold, i.e. the character distributions, is induced by the Oracle model. Training a new network with the generated labels may lead to performance degradation if this manifold cannot be preserved to some extend. To this end, the new network is initialized with the Oracle weights. Thus, its initialization is already a local optimum with respect to the training set, which encodes information about that set's character manifold. Learning rate selection is crucial in order to attain a well-performing adapted network. Small learning rates could trap the network to the initial local optimum of the Oracle, while large learning rate could, as usual, lead to convergence issues. In practice, we used 5× larger learning rate compared to the one used for the initial Oracle, proven to be capable of discovering nearby local optima corresponding to adapted manifolds.

Generated Labels and Losses: Considering the unique aspects of the handwritten text, we adopt the CTC loss along with a greedy decoding on the output of the Oracle in order to generate the corresponding labels. Therefore, a text string is generated for each text line, following the typical HTR training logic. An alternative approach to the CTC-based strategy is the usage of the per-column (of the RNN input feature map) character probability scores, as produced by the softmax operation over the output of the network. Since we can generate per-column character predictions using the Oracle network, such formulation can be addressed by a cross-entropy loss, minimizing these predictions.

As misclassified data are harmful for using with the transduction cycle, previous works have opted to use a criterion that selects only part of the data for retraining [4]. In this work, we avoid using a hard decision threshold, and instead we use a weighted version of the aforementioned loss based on the confidence of the Oracle predictions. This way we take into account possible unseen writing styles which score poorly with the Oracle model.

Considering the CTC loss, we cannot straightforwardly define a per-character weight. Nevertheless, we can generate a holistic score over the entire text line using the CTC loss and the produced decoding/label. Consequently, the new model will tend to be optimized with respect to the more confident decodings, while less confident decodings will contribute less to the total. In order to incorporate these scores into a weighted average formulation (considering batches of line images, as feed to the optimized) we apply a softmax operation to them. Given a set (batch) of images $\{I_i\}$ and their corresponding decodings $\{s_i\}$ $(i = 1, \dots, N)$ and the networks $h(\cdot)$ and $h_o(\cdot)$, which correspond to the training network and the Oracle network respectively, the weighted CTC formulation can be expressed as follows:

$$a_i = L_{CTC}(h(I_i), s_i) \tag{1}$$

$$w_i = \frac{e^{-c \cdot a_i}}{\sum_i e^{-c \cdot a_i}} \tag{2}$$

$$\text{loss} = \sum_i w_i \cdot L_{CTC}(h_o(I_i), s_i) \tag{3}$$

The c parameter of the softmax operation defines the divergence of the weights. We select a value of 0.1 in order to avoid extreme values which resemble a hard thresholding operation.

The weighted version of cross-entropy variant is rather intuitive, since each per-column prediction already has a clear-defined Oracle output probability score, which is used as a weight. Scaling is therefore performed on a per-column instead of a per-datum basis.

Iterative Re-training: If training with generated labels produced by the Oracle can successfully improve the network's performance, it is natural to extend the rationale by simply iteratively performing the same scheme in order to further increase performance. Specifically, the proposed method is proposed in training cycles. During each cycle the Oracle network is fixed and used only to produce the labels. At the end of the cycle the new model is fully trained and, ideally, has better performance than the Oracle. Therefore the Oracle is replaced by the trained model and a new model is trained during the next cycle using the enhanced Oracle network. This iterative procedure is executed until a predefined number of cycles or until no further loss improvement is observed. The entire proposed procedure is depicted in Figure 2.

Additionally, the initialization of the new network to be trained at the start of each cycle may affect this iterative approach. At the start of the whole procedure, the network to-be-trained is initialized with the same weights as the Oracle. Similarly, one may choose to have both Oracle and the new network initialized with the same parameters at each cycle. Typically, according to this strategy, the new network retains the weights of the previous cycle and thus no initialization is applied. However, we proposed a different approach; at the start of each cycle the network to-be-trained is initialized according to the initial Oracle, trained on the train set. The reasoning behind this choice is related to the manifold properties that we already mentioned. Specifically, if we continuously shift the initial underlying manifold by using as starting point the newly trained weights, we may considerably diverge from the initial distribution of the characters and lose important information from the train set. Therefore, if we consider the initial pre-trained network as the starting point of our method, we can converge to nearby solutions with similar properties, while using the updated labels of the iterative process. In other words, the proposed initialization acts as an implicit constraint on the structure of the final solution with respect to the character manifold on the initial train set.

Connection to Knowledge Distillation: The main idea, i.e. (partially) retaining the initial behavior of the network while training another network, closely resembles knowledge distillation approaches [8]. In fact, identical to our approach, distillation includes a pair of models: the teacher (corresponds to the Oracle) and the student model. Distillation consists of two losses, the task related one and the distillation loss, typically defined as the Kullback-Leibler (KL) divergence:

$$KL(p|q) = \sum_i p_i \log \frac{p_i}{q_i} \tag{4}$$

where p_i and q_i are the softmax activations at the output of the models. Typical distillation methods scale the outputs, before the softmax operation, by a temperature factor $1/T$ that "softens", when $T > 1$, the comparison of the two distributions. The distillation loss aims to bring the distributions of the models' outputs as close as possible.

The per-column weighting variant of the proposed method can be viewed as a distillation loss, since cross entropy is closely related to KL divergence: $H(p, q) = \sum_i p_i \log q_i$. Contrary to distillation logic, we want to mine existing, yet hidden, knowledge and not retain the Oracle's output distribution. Therefore the cross entropy approach is a hard-assignment variant of distillation, or equivalently, very small temperature factor values ($T \to 0$) are considered. As an extension, the CTC version of the proposed algorithm can be viewed as an sequence-wise alternative of a distillation loss[3].

4 Experimental Evaluation

Evaluation of our methods is performed on two widely used datasets, IAM [15] and RIMES [7]. The ablation study, considering different settings of the proposed methodology, is performed on the challenging IAM dataset, consisted of handwritten text from 657 different writers and partitioned into writer-independent train/test sets.

All experiments follow the same setting: line-level recognition using a lexicon-free greedy CTC decoding scheme. Character Error Rate (CER) and Word Error Rate (WER) metrics are reported in all cases (lower values are better).

The pre-processing steps, applied to every line image, are: 1. All images are resized to a resolution of 128×1024 pixels. Initial images are padded (using the image median value, usually zero) in order to attain the aforementioned fixed size. If the initial image is greater than the predefined size, the image is rescaled. The padding option aims to preserve the existing aspect ratio of the text images. 2. A simple global affine augmentation is applied at every image, considering only rotation and skew of small magnitude in order to generate valid images. Additionally, elastic net deformations are also applied, in order to simulate local spatial variation. 3. Each line transcription has spaces added at the beginning and at the end, i.e. "He rose from" is changed to " He rose from ". This operation aims to assist the system to predict the marginal spaces that exist in the majority of the images. [4]

Training is performed in two stages. *1) Training the initial Oracle, $t = 0$:* training for 240 epochs with cosine annealing and restarts every 40 epochs [13], with learning rate 0.001. *2) Transductive learning cycle:* each cycle consists of 20 epochs, while the learning rate is initialized at 0.005 and gradually reduced according to cosine annealing. We set an increased learning rate compared to the initial network in order to discover neighboring local optima and not be constrained to the one found by the initial training.

In the following, we explore different settings of the proposed self-training system, considering a variety of generalization schemes, losses and optimization strategies. To further understand the capabilities of the proposed method, we also consider sub-optimal pre-trained networks, trained only on a subset of the initial train dataset. Finally, we compare our approach to other methods that attain state-of-the-art accuracy on either the IAM and RIMES datasets.

4.1 Ablation Study

The impact of three different optimization aspects is explored for the case of a single-cycle self-training step.

[3] A detailed analysis on sequence-level knowledge distillation has been presented in [10].

[4] Note that during evaluation we do not take into account these added spaces, as this would bias the result in favour of our method.

Strategy	CER %	WER%
Initial	4.55	15.71
None	4.51	15.47
Dropout	4.41	15.24
Augmentation	4.29	14.99
Dropout + Augmentation	4.24	14.75

(a)

	CER %	WER%
CNN Backbone	4.37	15.04
Recurrent Head	4.48	15.54
Whole Network	4.24	14.75

(b)

Table 1. Impact of (a) generalization strategies and (b) part-optimization of the proposed transductive method.

Impact of Generalization Strategies: We have first experimented on how the proposed transductive method combines with two different strategies to boost the model ability to generalize to new data. The strategies considered are Dropout and Augmentation, and the corresponding models are compared on Table 1(a). Conceptually, the proposed algorithm may benefit from an overfitting to the test data in order to fully utilize the existing data and thus generalization scheme may lead to worse performance. Nonetheless, from the results we can conclude that these strategies together with the proposed method combine well. Especially augmentation provides a considerable boost in performance, hinting that we may discover correct labels when slightly transforming the image, while we can better formulate the underlying character manifold/distribution of the test data by adding more (synthetically generated) instances.

Impact of Optimizing Parameters: Re-training the whole network may in principle lead to worse-performing local optima, especially in the context of transductive learning. Keeping some layers fixed may act as a constraint in order to retain desirable properties from the initial network. In order to check whether one network component is more robust than the rest in this respect, we consider re-training either the CNN-backbone part or the recurrent head part, while keeping the other part fixed. Error rate results are presented in Table 1(b). Results suggest that the best option is to leave no part of the model frozen.

Impact of type and weighting of the loss function: As described in section 3.2, we use a weighting scheme when using Oracle labels. We have experimented in using each of the described alternatives: using CTC-based scores versus CE-based scores. Also, we compare against using equal weights to all Oracle labels. The results can be examined in Table 2. Weighting variants appear marginally better than their unweighted counterparts. More importantly, CTC-based approaches significantly outperform CE-based ones. This can be attributed to the fact that the CE method assigns a label to each output column, including transitional columns between two characters that may not correspond to a specific character. Weighting does not appear to solve this problem, neither preliminary experiments with predefined thresholds. On the other hand the CTC-based approach is better suited for the problem at hand and it can be effective even when the unweighted version is considered.

Loss	CER%	WER%
Initial	4.55	15.71
CTC	4.24	14.75
CE	4.37	15.16
wCTC	4.23	14.74
wCE	4.35	15.09

Table 2. Comparison of weighting variants. CTC: Results using a CTC-based loss. CE: Results using a CE-based loss. wCTC: Results using a CTC-based loss with each datum loss weighted according to the Oracle CTC decoding score. wCE: Results using a CE-based loss, with each feature map column loss weighted according to the Oracle CE output.

4.2 Effect of using multiple iterations

Theoretically, after performing a self-training cycle, we acquire a better-performing network, ideal for acting as the new Oracle for an upcoming cycle, as we have already described in Section 3.2 The ability of multiple iterations of the proposed self-training approach (wCTC variant) to further increase the network's performance is observed in Table 3.

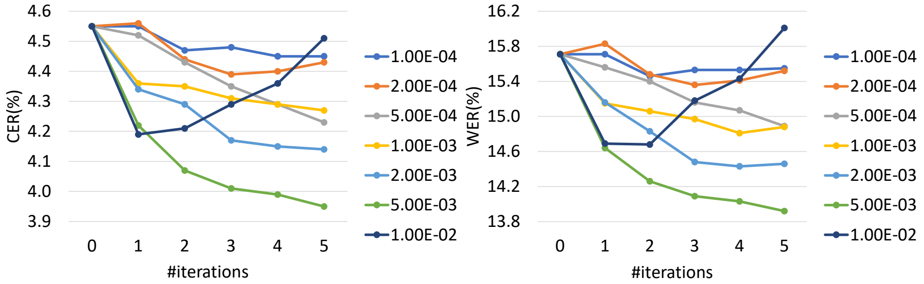
Iteration	CER %	WER%
Initial	4.55	15.71
1	4.24	14.75
2	4.07	14.26
3	4.01	14.09
4	3.99	14.03
5	3.95	13.92

Table 3. HTR performance using different numbers of transductive cycle iterations.

We further investigate the impact of the initial learning rate at the start of each cycle. The results are presented in Figure 3. As suggested, selecting small learning rates provides a minor boost in performance. Note that both values of $5 \cdot 10^{-3}$ and 10^{-2}, provide a similar boost after first cycle. However, the latter seems to diverge from meaningful solutions in the upcoming cycles.

4.3 Using a suboptimal Oracle

Transductive learning has been proved to be a good option before, especially when data is very limited [9, 6]. In order to test how the proposed method fares under a limited-resource paradigm, we have tested the method with Oracles that were trained with only a small percentage of the originally available labelled training data. We have trained our Oracle using a number of IAM training lines that correspond to percentages equal to 10%, 25%, 50%. Results can be examined



(a) (b)

Fig. 3. Impact of learning rate for the iterative procedure. Different metrics are depicted in (a) and (b).

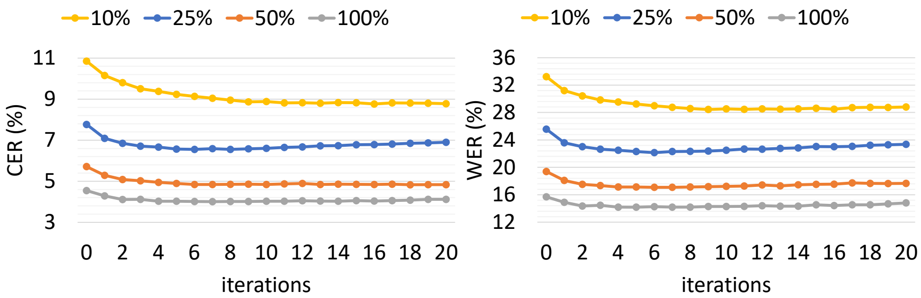
in Table 4 and Figure 4. In all cases, the proposed method is shown to improve results, steadily up to a margin of $1-2\%$ for CER and $3-5\%$ for WER.

percentage=	10%		25%		50%	
Iteration	CER %	WER%	CER %	WER%	CER %	WER%
Initial	10.85	33.24	7.77	25.57	5.72	19.40
1	10.02	30.95	7.22	24.22	5.26	17.99
2	9.65	29.96	6.69	22.55	5.07	17.42
3	9.36	29.40	6.63	22.45	4.96	17.22
4	9.18	28.96	6.55	22.37	4.91	17.05
5	9.04	28.72	6.52	22.24	4.87	16.88

Table 4. Tests under conditions where the Oracle has trained on limited data. Percentages on the top row correspond to percentages of total training lines of the IAM dataset that where used to train the Oracle. Per-row iterations and scores correspond to different iterations of the transduction learning cycle.

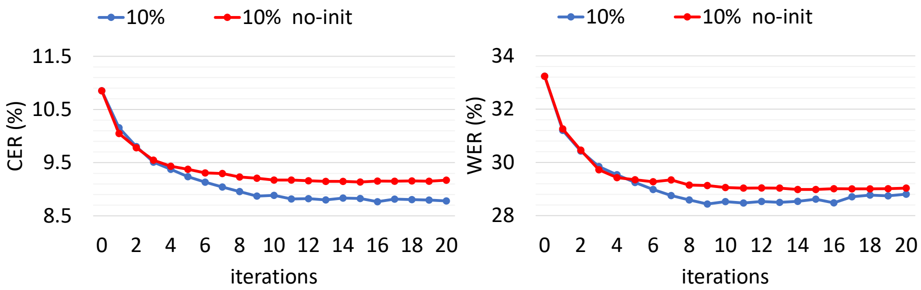
Figure 4 provides additional insights on the iterative approach and specifically its behavior for a large number of transductive cycles. The performance gain is quickly saturated, while small performance degradation can be observed for over 10 iterations. Such degradation is expected, since the Oracle may produce many missclassified predictions for a specific character, but the degraded networks still retain significantly better results than the initial network trained on the train set. The case of 10% appears to be continuously improving up to 10 iterations, unlike the other cases, due to the larger capability of improvement.

Furthermore, we evaluate the proposed initialization scheme, described in section 3.2, at the start of each transductive cycle. Initializing with the pre-trained network or retaining the weights (*no-init*), appear to have similar performance for most of the cases. Their difference can be detected at the more demanding 10% case as shown in Figure 5, where the proposed method saturates slower than the typical no initialization scheme.



(a) (b)

Fig. 4. Behavior of HTR performance when considering a large number of iterations. Two different metrics are reported: (a) CER and (b) WER.



(a) (b)

Fig. 5. Impact of initialization strategy at each cycle. Both metrics are considered: (a) CER and (b) WER.

Finally, the per-character improvement is shown in Figure 6 for two cases: 10% and 100% (whole train set). Specifically we report the number of corrected characters after 5 cycles of the proposed methodology. Negative values correspond to worse performance compared to the initial pre-trained network. We can observe that, as expected, large number of corrected characters are reported for the 10% case. Nevertheless, there are characters that may further deteriorate their performance. A characteristic case is the character i, where a notable decrease in performance is observed. The sub-optimal initial network may often confuse i with characters of similar appearance, such as j and l. Such behavior can be intensified by the proposed iterative procedure.

4.4 Comparison with the State of the Art

Finally, we have compared the proposed HTR model with the proposed learning scheme against state-of-the-art methods in the literature. The variant used for the proposed method uses weighted CTC loss and 5 retraining iterations. Comparative results can be examined in Table 5. We can observe that the proposed CNN-RNN model is already very competitive, with or without the transduction scheme being applied. The proposed HTR architecture, combined with the

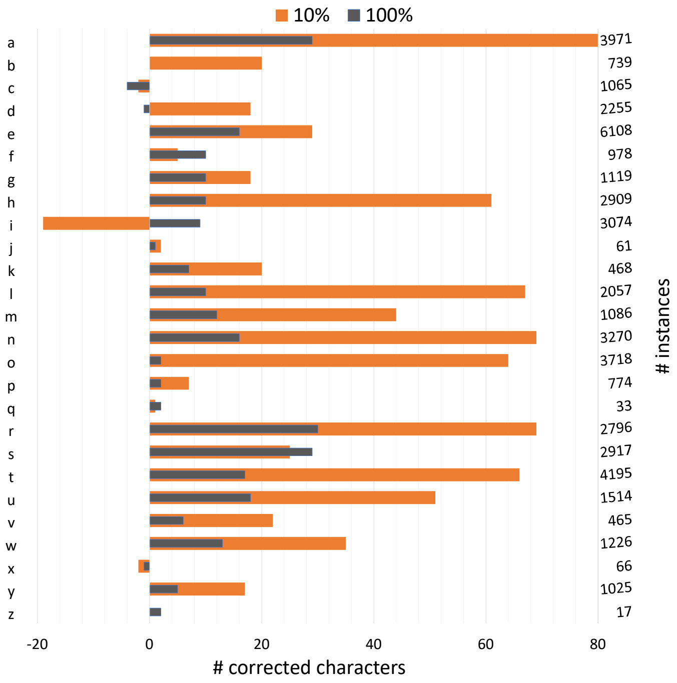


Fig. 6. Per-character analysis of improvements using the proposed method. Results for two settings are compared: using all of the IAM training set or a part of it (10%), in order to simulate the case where training data are limited. The horizontal axis denotes number of extra corrected characters using the proposed method versus not using it. The vertical axis denotes each different character and number of total test instances.

transduction scheme leads to results that are on par with the state of the art in both IAM and RIMES.

5 Conclusion

We have proposed a transductive learning method for Handwriting Recognition models, and we have shown that by combining it with a competitive CNN-RNN architecture the model reaches state-of-the-art recognition results. By using repeated transduction iterations we have shown that results can further improve. Furthermore, we have compared two weighting methods for more precise tuning of the reference model loss. Regarding future work, a possible research perspective could be exploring transduction in terms of a projection between training and test set geometry, or considering restating the method within a probabilistic framework [21, 23]. Also, we envisage testing the method on scripts with more limited data available than latin [5, 22].

	IAM		RIMES	
Architecture	CER %	WER %	CER %	WER %
Puigcerver [18]	6.2	20.2	2.6	10.7
Krishnan et al. [11]	9.78	32.89	-	-
Dutta et al. [3]	5.8	17.8	5.07	14.7
Chowdhury and Vig [1]	8.10	16.70	3.59	9.60
Michael et al. [16]	4.87	-	-	-
Tassopoulou et al. [24]	5.18	17.68	-	-
Yousef et al. [27]	4.9	-	-	-
Markou et al. [14]	6.14	20.04	3.34	11.23
Xiao et al. [26]	3.97	12.90	1.56	12.61
proposed w/out transduction	4.55	15.71	3.16	11.09
proposed w/ transduction, @5 iter	3.95	13.92	2.90	10.17

Table 5. Performance comparison for IAM/RIMES datasets.

References

1. Chowdhury, A., Vig, L.: An efficient end-to-end neural model for handwritten text recognition (2018)
2. Cozman, F.G., Cohen, I., Cirelo, M.: Unlabeled data can degrade classification performance of generative classifiers. In: Flairs conference. pp. 327–331 (2002)
3. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Improving CNN-RNN hybrid networks for handwriting recognition. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 80–85. IEEE (2018)
4. Frinken, V., Bunke, H.: Self-training strategies for handwriting word recognition. In: Industrial Conference on Data Mining. pp. 291–300. Springer (2009)
5. Giotis, A.P., Sfikas, G., Nikou, C., Gatos, B.: Shape-based word spotting in handwritten document images. In: 13th International conference on document analysis and recognition (ICDAR). pp. 561–565. IEEE (2015)
6. Granet, A., Morin, E., Mouchère, H., Quiniou, S., Viard-Gaudin, C.: Transfer learning for a letter-ngrams to word decoder in the context of historical handwriting recognition with scarce resources. In: 27th International Conference on Computational Linguistics (COLING). pp. 1474–1484 (2018)
7. Grosicki, E., Carre, M., Brodin, J.M., Geoffrois, E.: Rimes evaluation campaign for handwritten mail processing (2008)
8. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
9. Keret, S., Wolf, L., Dershowitz, N., Werner, E., Almogi, O., Wangchuk, D.: Transductive learning for reading handwritten tibetan manuscripts. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 214–221. IEEE (2019)
10. Kim, Y., Rush, A.M.: Sequence-level knowledge distillation. arXiv preprint arXiv:1606.07947 (2016)
11. Krishnan, P., Dutta, K., Jawahar, C.: Word spotting and recognition using deep embedding. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS). pp. 1–6. IEEE (2018)
12. Lefakis, L.: Transductive Learning for Document Classification and Handwritten Character Recognition. Ph.D. thesis, Citeseer (2008)

13. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
14. Markou, K., Tsochatzidis, L., Zagoris, K., Papazoglou, A., Karagiannis, X., Symeonidis, S., Pratikakis, I.: A convolutional recurrent neural network for the handwritten text recognition of historical greek manuscripts. In: International Workshop on Pattern Recognition for Cultural Heritage (PATRECH) (2020)
15. Marti, U.V., Bunke, H.: The iam-database: an english sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition **5**(1), 39–46 (2002)
16. Michael, J., Labahn, R., Grüning, T., Zöllner, J.: Evaluating sequence-to-sequence models for handwritten text recognition. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1286–1293. IEEE (2019)
17. Mühlberger, G., Seaward, L., Terras, M., Oliveira, S.A., Bosch, V., Bryan, M., Colutto, S., Déjean, H., Diem, M., Fiel, S., et al.: Transforming scholarship in the archives through handwritten text recognition. Journal of documentation (2019)
18. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 67–72. IEEE (2017)
19. Retsinas, G., Louloudis, G., Stamatopoulos, N., Gatos, B.: Efficient learning-free keyword spotting. IEEE transactions on pattern analysis and machine intelligence **41**(7), 1587–1600 (2018)
20. Retsinas, G., Sfikas, G., Gatos, B.: Transferable deep features for keyword spotting. In: Multidisciplinary Digital Publishing Institute Proceedings. vol. 2, p. 89 (2018)
21. Sfikas, G., Gatos, B., Nikou, C.: Semicca: A new semi-supervised probabilistic cca model for keyword spotting. In: 2017 IEEE International Conference on Image Processing (ICIP). pp. 1107–1111. IEEE (2017)
22. Sfikas, G., Giotis, A.P., Louloudis, G., Gatos, B.: Using attributes for word spotting and recognition in polytonic greek documents. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 686–690. IEEE (2015)
23. Sfikas, G., Nikou, C., Galatsanos, N., Heinrich, C.: Majorization-minimization mixture model determination in image segmentation. In: CVPR 2011. pp. 2169–2176. IEEE (2011)
24. Tassopoulou, V., Retsinas, G., Maragos, P.: Enhancing handwritten text recognition with n-gram sequence decomposition and multitask learning. arXiv preprint arXiv:2012.14459 (2020)
25. Tensmeyer, C., Wigington, C., Davis, B., Stewart, S., Martinez, T., Barrett, W.: Language model supervision for handwriting recognition model adaptation. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 133–138. IEEE (2018)
26. Xiao, S., Peng, L., Yan, R., Wang, S.: Deep network with pixel-level rectification and robust training for handwriting recognition. SN Computer Science **1**(3), 1–13 (2020)
27. Yousef, M., Hussain, K.F., Mohammed, U.S.: Accurate, data-efficient, unconstrained text recognition with convolutional neural networks. Pattern Recognition **108**, 107482 (2020)
28. Zhou, Z.H.: A brief introduction to weakly supervised learning. National science review **5**(1), 44–53 (2018)