Reducing the Manual Annotation Effort for Handwriting Recognition Using Active Transfer Learning

Eric Burdett
Brigham Young University

Follow this and additional works at: https://scholarsarchive.byu.edu/etd
Part of the Physical Sciences and Mathematics Commons

BYU ScholarsArchive Citation
https://scholarsarchive.byu.edu/etd/9258

This Thesis is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of BYU ScholarsArchive. For more information, please contact ellen_amatangelo@byu.edu.
Reducing the Manual Annotation Effort for Handwriting Recognition

Using Active Transfer Learning

Eric Burdett

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of
Master of Science

Mark Clement, Chair
Bryan Morse
Dan Ventura

Department of Computer Science
Brigham Young University

Copyright © 2021 Eric Burdett
All Rights Reserved
ABSTRACT

Reducing the Manual Annotation Effort for Handwriting Recognition Using Active Transfer Learning

Eric Burdett
Department of Computer Science, BYU
Master of Science

Handwriting recognition systems have achieved remarkable performance over the past several years with the advent of deep neural networks. For high-quality recognition, these models require large amounts of labeled training data, which can be difficult to obtain. Various methods to reduce this effort have been proposed in the realms of active and transfer learning, but not in combination. We propose a framework for fitting new handwriting recognition models that joins active and transfer learning into a unified framework. Empirical results show the superiority of our method compared to traditional active learning, transfer learning, or standard supervised training schemes.

Keywords: handwriting recognition, active learning, transfer learning
ACKNOWLEDGMENTS

First and foremost, I would like to thank my wife, Erika, for the support she has given me to accomplish this work. Her encouragement through the frequent struggles of a graduate program has been invaluable. Additional thanks goes to my advisor, Mark Clement, for the amazing opportunity to perform research in the handwriting lab. His faith in me has provided countless opportunities for personal, academic, and career growth. Special thanks also goes out to the Ancestry Data Science Team for providing challenging projects and funding to encourage my research. The collaboration and feedback from Ancestry team members, Yen Yun Yu, Gopalkrishna Veni, and Stanley Fujimoto has been deeply appreciated.
Table of Contents

1 In Preparation: Reducing the Manual Annotation Effort for Handwriting Recognition Using Active Transfer Learning 1
Chapter 1

In Preparation: Reducing the Manual Annotation Effort for Handwriting Recognition Using Active Transfer Learning

This manuscript has not yet been accepted for publication.
Reducing the Manual Annotation Effort for Handwriting Recognition using Active Transfer Learning

Eric Burdett and Mark Clement

Brigham Young University, Provo UT 84602, USA
eric.burdett@byu.edu, clement@cs.byu.edu

Abstract. Handwriting recognition systems have achieved remarkable performance over the past several years with the advent of deep neural networks. For high-quality recognition, these models require large amounts of labeled training data, which can be difficult to obtain. Various methods to reduce this effort have been proposed in the realms of active and transfer learning, but not in combination. We propose a framework for fitting new handwriting recognition models that joins active and transfer learning into a unified framework. Empirical results show the superiority of our method compared to traditional active learning, transfer learning, or standard supervised training schemes.

Keywords: handwriting recognition · active learning · transfer learning.

1 Introduction

Handwriting recognition has received great attention recently with its promise to reduce the human effort of manually transcribing handwritten text. Although great strides have been made to transcribe handwritten records using deep learning and statistical language models, they are still somewhat limited due to the variability in writing style and the inability to generalize well across datasets. Thus, when a new set of handwritten records requires recognition, the common solution is to manually annotate new training samples from the target domain. This manual process is not a trivial one, especially when samples are required in large quantities. Expert knowledge is often required, the process is time-consuming, and ensuring consistency among annotators is a common issue. These difficulties have, in part, resulted in many digitized records remaining without transcriptions. This results in the inability to easily search historical records for use in genealogical research and prohibits researchers from examining historical patterns and trends that could help solve significant societal issues. Any reduction in the manual annotation effort required to train these models would be particularly valuable as it would allow recognition to be applied to more records and provide additional searchable historical collections for the community.

A significant amount of research has been conducted in attempting to reduce this manual annotation effort. Many of the proposed solutions revolve around
transfer learning and domain adaptation techniques [18, 19, 22], synthetic data creation [2, 5, 8, 9, 22], self-supervision [1, 22], and active learning [26, 31]. Although these methods have been explored in depth independently, there has been little research combining these methods into a single framework.

The problem with using transfer learning techniques independently (without additional labeled data from the target set) is that recognition accuracy is usually sacrificed. One may avoid labeling additional data, but a high-quality recognition model usually cannot be obtained. Even though much of the previous research in transfer learning methods for recognition is valuable, and can spawn new research ideas, using these methods independently often does not have much practical use, due to the degraded performance. It seems that there is rarely a substitute for good labeled training data from the target domain. Although it is common for researchers to fine-tune pre-trained models on additional training data from the target domain, there is not a well-defined methodology, nor are there many documented experiments that show the impact with respect to handwriting recognition specifically. Without this well-defined process, it can be difficult to know which pre-trained model to use, which instances to label from the target domain, and how much additional training data is required to obtain good performance.

Active learning has been considered for training handwriting recognition models, but only with a narrow view [26, 31]. The small set of active learning papers in this space considers only a single dataset that is used to train a model from scratch. Despite modest labeling reductions, the annotation effort can still be quite large. With the overhead of training new annotators and encouraging consistency with labeling, a 10-20% reduction in labeled samples may not make as big of a difference as originally hoped.

Combining active and transfer learning allows us to utilize previously learned knowledge from similar handwriting styles while also providing a method for selecting the most important samples to label that will help overcome the weaknesses of the transferred model. In this work, we propose a clear and methodical approach for fitting new handwriting recognition datasets by proposing an active transfer solution. This approach minimizes the annotation effort and provides tunable accuracy depending on the number of samples that are labeled. Empirical results show the effectiveness of the active transfer framework compared to traditional active learning, transfer learning, and standard supervised training schemes.

2 Related Work

A significant amount of research has been conducted to reduce the manual annotation effort for handwriting recognition. These methods usually fall under the realm of transfer learning, active learning, synthetic data creation, and self-supervision. We will briefly discuss each of these methods and consider the previously proposed applications for recognition.
2.1 Transfer Learning

Transfer learning is a broad term used to describe the concept of utilizing past knowledge learned on previous tasks for the current task. In the simplest case, this involves using a model trained on one dataset and using it for another. Fine-tuning a pre-trained model with additional labeled data from the target domain is also a fairly common computer vision practice [33].

Other transfer learning techniques involve training mechanisms that use labeled data from a source domain and unlabeled data from the target domain and conditions the model to be more receptive to target data. For example, Domain Adversarial Networks [11] propose a training scheme to encourage domain confusion and consist of a domain classification network and a gradient-reversal layer. When used in conjunction with standard supervised training in the source domain, the process encourages the feature-extraction layers to become domain-invariant. This method was applied to handwriting recognition where a synthetic dataset was used as the source domain and real handwritten text as the target domain [19]. The results were impressive with substantial reductions in error rates. However, error rates were still too high to be used in practice. Others applying domain adversarial networks to handwriting recognition include [22] and [18], with the latter seeking to disentangle content from style in handwritten images.

Image-to-Image translation is also a common solution when attempting to make recognition models more generalizeable. These solutions usually use some form of generative adversarial network (GAN) [14] or style-transfer [12] variant to perform the mapping. A Conditional Sliced Wasserstein GAN was used [20] to map difficult to recognize handwriting into an easier domain. The results were impressive, but the framework requires a dataset with paired source and target domain images for training. Cyle-consistent GANs [34] introduced an unsupervised approach to mapping between image domains. This has been applied to handwriting to make a synthetic Chinese dataset look more like the target domain [5]. It was also attempted for Tibetan handwriting [22]. A style-transfer approach was also proposed to map between text styles [13]. The method worked well for scene-text, but struggled with handwriting, as the mappings seemed to focus more on features such as background color, stroke size, and textures rather than modeling the variation in handwriting style.

2.2 Synthetic Data Creation

In domains where HWR datasets are limited, many have resorted to creating synthetic datasets [5, 22]. This often includes generating images using machine printed text where the given labeling for an image can be controlled. Various data augmentation techniques are then applied to images in the dataset and subsequently used for training. Some have gone to great lengths to show the variation in handwriting style that is exhibited when using their augmentation technique [28, 32]. These augmentation methods usually involve applying elastic
distortions rather than classic affine transformations. Others have also attempted to perform data augmentation at the feature level using an external network [3].

Synthetic data creation has become more sophisticated in recent years with greater adoption of GANs, and have been applied to handwriting recognition by [2, 9, 8]. These GANs usually work by conditioning the network on a word string which controls the model output; thus allowing a significant amount of labeled data to be added to the training pool. Recently, [8] set up an architecture similar to the previous solutions, but used an additional network for extracting the style of an image which is then applied to the GAN output. This allows for synthetic images to be created using the style of a given input image. This is somewhat reminiscent of the popular Style-GAN [21].

2.3 Self-Supervision

Some have found success using a technique called self-supervision in which a pre-trained model makes predictions on data from the target domain [10, 22]. If the model is confident enough about its prediction of a given image, the predicted label and image are then used as actual training data. Another common self-supervision technique in computer vision is SimCLR [7] and works solely on unlabeled data. This pre-training technique works by extracting features from two augmented variations of an image and taking a contrastive loss between them. This forces the feature representations of the two augmented images to be pulled closer together, while all other feature representations from different images are pushed further apart. This idea was adopted to handwriting recognition and dubbed SeqCLR [1], which allows the contrastive framework to work better for domains with sequential outputs. Their proposed solution was shown to achieve similar error rates with less training data and state-of-the-art performance when all available training data is used. While still a good method for reducing the annotation effort, a greater reduction may be achieved by combining this with active learning.

2.4 Active Learning

In contrast to selecting random instances for a training set, active learning is the process of selecting the most informative samples that will lead to superior model performance. The key element to active learning is the selection of an uncertainty function, $\phi$ that will determine which samples will be annotated. Common examples of $\phi$ include using the magnitude of the most probable labeling, the margin between the first and second most probable labelings, and Shannon entropy [31].

Active learning has not been a well studied topic in the realm of handwriting recognition and is limited to a small set of papers. [31] proposed using n-best list entropy and the global entropy reduction for the uncertainty function. Using n-best list entropy, they were able to reduce the annotation effort by eighteen percent [31]. Most recently, [26] took an active learning approach by using derivational entropy to calculate $\phi$. Their results indicated a small reduction of
annotation effort, but their recognition error rates were still high due to their use of hidden markov models rather than state-of-the-art deep neural networks. In both cases, the authors considered active learning in the context of a single dataset without utilizing previously trained models, labeled data from other datasets, or applying any transfer learning techniques.

2.5 Active Transfer Learning

To the best of our knowledge, the use of active learning and transfer learning in a single framework has not been studied for handwriting recognition; however, it has been studied in other areas of computer vision and natural language processing. To avoid the cold-start problem often seen in standard active learning, [17] used a set of pre-trained model weights to initialize the active learner. Others have built upon this framework by using domain separation classifiers and clustering methods for imputing labels to provide slight increases in accuracy with the same amount of data [6, 25, 27].

In most cases, active learning and transfer learning are performed in two separate distinct steps. More recently, however, [29] developed a loss function integrating both active and transfer learning techniques. This framework uses a domain adversarial network [11] with the addition of labeled data from the target domain being actively selected and trained in a supervised manner concurrently with domain adversarial training.

To the best of our knowledge, Active Transfer Learning has not been attempted specifically for handwriting recognition.

3 Methodology

We propose a framework for fitting handwriting recognition models to new datasets by combining active and transfer learning. Generally, transfer learning and active learning are used as individual methods for reducing the annotation effort to train recognition models. Rather than starting with a random weights initialization, we use a set of weights pre-trained on another dataset. Then, instead of randomly selecting images to label, we actively select images that will be most helpful for training. The labeled images are added to the pool of labeled target data and the model is subsequently trained. This process of actively selecting labels and training the model with the entire pool of labeled data is repeated iteratively until the desired accuracy is achieved or until the model has fully converged and no improvement is observed.

The proposed framework is designed to be modular, where pre-trained weights and active learning algorithms can easily be inserted and replaced. The following provides a high-level overview of each step of the framework and the implementation we used to achieve our results. Figure 1 provides a visual representation of the ideas presented in this section.
Fig. 1. Proposed system architecture for fitting a new dataset. We start by determining which of our pre-trained models will be best suited for the new dataset. The pre-trained model along with unlabeled target data are given to the active learner for sample selection. The chosen samples are annotated and added to the pool of labeled target data. The model is trained and immediately evaluated. If determined after training that the model has achieved sufficient recognition accuracy, we are finished. Otherwise, we iteratively repeat the process until the desired accuracy is obtained.
3.1 Model Weights Initialization

The first step in our framework is determining which set of pre-trained model weights will provide the best initialization for the model that will be fine-tuned on data from the target domain. We propose two options. First, we can use the weights from models that have been trained on other datasets that are somewhat similar to the target domain. Second, we can use a pre-training, self-supervision technique such as SeqCLR [1] to obtain our initial set of model weights. These two options are described below.

Pre-trained Recognition Models If multiple models exist that have been trained on data that are somewhat similar to the target domain, we can evaluate these models in two ways. First, we can evaluate the weights using a test set already created for the target domain. If a test set has not been created, we can use the average confidence score given by the recognition model when provided with a random sample of images from the target set.

For a confidence measure, we make use of the Connectionist Temporal Classification (CTC) loss function [15], which is used extensively when training handwriting recognition models. CTC provides a loss between the model’s output, which is a two-dimensional matrix representing the probability of each character occurring for each timestep in the sequence, and the provided ground truth. For generating a confidence score, we can utilize this same loss function by using the model’s best prediction as the ground truth, which is commonly the best-path (greedy) decoding of the model’s output [15]. We can then transform the CTC loss value, given as the negative log likelihood, back to a probability by applying an exponential and removing the negative. Equation 1 provides the probability that the model’s prediction is correct with $L$ being the result of CTC loss when providing the best-path decoding of the model’s prediction as the ground truth.

$$p = e^{-L}$$ (1)

Using this equation, we can provide a confidence score for how the model performs on the dataset generally. We do this by taking $n$ random samples from the dataset and calculating the mean of all confidence scores. The model that provides the highest mean confidence over a random sample will be used. The weights from this model will then be used as the starting initialization for the model in the active transfer framework.

SeqCLR Pre-training If no pre-trained recognition models exist from similar domains, model weights can be pre-trained using the SeqCLR framework [1]. For all experiments with SeqCLR, we use the window-to-instance mapping function and train with a learning rate of 0.001 for an unlimited number of epochs and apply early stopping with a patience of 10 epochs. We train using a batch size of 512 and use a variety of sequence-preserving data augmentations, many of which are described in [1]. Appendix A provides a more detailed description of the augmentations used.
The contrastive loss is applied on feature maps directly before the final dense layer of the recognition model. Due to the architecture of the recognition model already containing contextualized features, we opted to forego the inclusion of an additional projection head, and apply the contrastive loss directly after the instance mapping function as described in [1]. After the model has been sufficiently trained, we use this set of model weights as the starting initialization for the active transfer method.

3.2 Active Learning Sample Selection

The active learning step involves making predictions on samples from the unlabeled target set and scoring them with an uncertainty function, \( \phi \). Although various uncertainty functions can be inserted here, we choose to use standard entropy adapted for use with Connectionist Temporal Classification [15]. Because we can obtain a probability score for each prediction as given in Equation 1, we can calculate the standard Shannon entropy to be used as the uncertainty function.

\[
\phi = -p \log(p)
\]  

Incorporating CTC loss as given in Equation 1 results in the following equation:

\[
\phi = -e^{-L} \log(e^{-L})
\]

The above equation simplifies to produce our final uncertainty function:

\[
\phi = e^{-L} \cdot L
\]

Using this criterion, we then take the top \( k \) images with the highest uncertainty scores to give to the annotator for labeling. These images are added to the set of labeled data in the target domain, which is subsequently used for standard supervised training. For all of our experiments, we use a value of \( k = 1000 \) and rather than performing the actual annotation, we simply retrieve the provided transcription from the training set of that particular dataset.

3.3 Supervised Training

With the labeled data given by the annotator, we train the model using standard supervised training with CTC loss [15]. Note that with each iteration of the active transfer learning process, we train with the entire pool of labeled target data. This pool is increased each iteration by the value of \( k \) as described in Section 3.2.

For each iteration, the model is trained for an unlimited number of epochs with early stopping used when there is no improvement on the validation set. The set of model weights that achieved the lowest loss on the validation set will be carried on to the next step of the framework. For all of our experiments, the value of patience for early stopping is set to 10. Note that performing early...
stopping requires a small subset of data from the target domain to be labeled beforehand. If this extra step of labeling is not possible, the model can be trained for a fixed number of epochs in each iteration. We resize all images to a height and width of (32, 128) and apply the random grid warp augmentation [32] during the training process.

**Recognition Model** For all of our experiments, we use the recognition model described in [30]. The model consists of an encoder with several gated convolutional layers [4] and a decoder containing several layers of bidirectional GRUs. The goal of this work is not necessarily to achieve state-of-the-art performance on certain benchmarks, but rather to show the impact of active transfer learning for handwriting recognition compared to the traditional active, transfer, and standard supervised learning methods. Any recognition model can be easily inserted into the active transfer framework.

### 3.4 Model Evaluation

Each iteration after the model has been trained, we evaluate its performance with a labeled test set in the target domain using the character error rate and word error rate. At times, it can be difficult to determine how much labeled data is needed to train a recognition model for a specific task. By evaluating the model each iteration, we can be sure to acquire enough labeled data to achieve the level of accuracy that is required for the task while not expending the effort and cost of labeling more data than is necessary. In essence, this step provides tunable accuracy, where the user can decide how much additional labeled data to provide depending on current model performance. Once the model has achieved an acceptable level of accuracy or the model’s improvement has plateaued, we can break from the loop and consider the active transfer training process complete. If the model is still converging and the model performance has not yet crossed an acceptable threshold, we can continue to iterate until that point is reached.

### 4 Results

To validate our methodology, we test our framework on the IAM [24], RIMES [16], and CVL [23] datasets. They were selected because they are somewhat similar in format, but vary slightly in style. All are modern datasets and contain black/blue handwriting with white backgrounds. All trained models are evaluated at the word-level on their respective test splits. More in depth descriptions of the datasets can be found in Appendix B. We evaluate each of the training setups using the standard character error rate and word error rate.

#### 4.1 Methods Comparison

To evaluate the active transfer framework, we compare the results of training a handwriting recognition model using each of the following formats:
The recognition models for the *Transfer* and *Active Transfer* methods were initialized with weights from a pre-trained model on the IAM and RIMES datasets respectively. For the RIMES dataset, the Active Transfer method achieved better error rates regardless of the number of labeled samples given for training. For the IAM dataset, the Active Transfer method achieved only comparable performance with the Active method likely because the pre-trained RIMES model provided very little benefit when training on IAM.
– **Scratch**: This is the typical supervised training method where the model is trained from scratch with a random weights initialization and a random selection of labeled training images.

– **Transfer**: The model weights are initialized from a pre-trained recognition model. Training images are randomly sampled from the target domain.

– **Active Scratch**: The model weights are trained from scratch with a random initialization, but data is actively selected from the target domain.

– **Active Transfer**: The model weights are initialized from a pre-trained recognition model. Training images are actively selected from the target domain.

When comparing these results, we observe the character error rate and word error rate when given the specified number of labeled samples. Figure 2 provides the results of the IAM and RIMES datasets across all four methods. The model for the RIMES dataset is initialized with a set of weights pre-trained on the IAM dataset. The model for the IAM dataset uses weights from RIMES.

**Table 1.** Best error rates and required samples to achieve fully-trained *Scratch* performance on the RIMES, IAM, and CVL datasets. This table is meant to show the reduction in labeled samples to achieve the same performance as a fully-trained *Scratch* model. For example, on the RIMES dataset, the *Active Transfer* method achieved an error rate of 5.20% with only 18,000 samples as opposed to 49,000 if the model was trained from *Scratch*. This results in a reduction of 31,000 samples. Note that the RIMES, IAM, and CVL datasets used pre-trained weights from the IAM, RIMES, and IAM datasets respectively.

<table>
<thead>
<tr>
<th>RIMES Dataset</th>
<th>Method</th>
<th>CER</th>
<th>WER</th>
<th>Samples(CER)</th>
<th>Samples(WER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
<td>5.20%</td>
<td>15.87%</td>
<td>49,000</td>
<td>49,000</td>
<td></td>
</tr>
<tr>
<td>Transfer</td>
<td>4.82%</td>
<td>15.19%</td>
<td>48,000</td>
<td>48,000</td>
<td></td>
</tr>
<tr>
<td>Active Scratch</td>
<td>4.97%</td>
<td>14.40%</td>
<td>29,000</td>
<td>29,000</td>
<td></td>
</tr>
<tr>
<td>Active Transfer</td>
<td><strong>4.33%</strong></td>
<td><strong>13.21%</strong></td>
<td><strong>18,000</strong></td>
<td><strong>20,000</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IAM Dataset</th>
<th>Method</th>
<th>CER</th>
<th>WER</th>
<th>Samples(CER)</th>
<th>Samples(WER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
<td>14.63%</td>
<td>29.96%</td>
<td>48,000</td>
<td>53,000</td>
<td></td>
</tr>
<tr>
<td>Transfer</td>
<td>15.06%</td>
<td>30.26%</td>
<td>&gt; 53,838</td>
<td>&gt; 53,838</td>
<td></td>
</tr>
<tr>
<td>Active Scratch</td>
<td><strong>13.76%</strong></td>
<td><strong>27.34%</strong></td>
<td><strong>32,000</strong></td>
<td>27,000</td>
<td></td>
</tr>
<tr>
<td>Active Transfer</td>
<td>14.05%</td>
<td>27.81%</td>
<td>35,000</td>
<td><strong>23,000</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CVL Dataset</th>
<th>Method</th>
<th>CER</th>
<th>WER</th>
<th>Samples(CER)</th>
<th>Samples(WER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scratch</td>
<td>18.38%</td>
<td>29.86%</td>
<td>83,000</td>
<td>83,000</td>
<td></td>
</tr>
<tr>
<td>Transfer</td>
<td>13.73%</td>
<td>29.78%</td>
<td>3,000</td>
<td>87,000</td>
<td></td>
</tr>
<tr>
<td>Active Scratch</td>
<td>15.78%</td>
<td>25.62%</td>
<td>17,000</td>
<td>15,000</td>
<td></td>
</tr>
<tr>
<td>Active Transfer</td>
<td><strong>13.62%</strong></td>
<td><strong>25.27%</strong></td>
<td><strong>3,000</strong></td>
<td><strong>12,000</strong></td>
<td></td>
</tr>
</tbody>
</table>

12
For the RIMES dataset, the Active Transfer method was the clear winner and achieved better error rates compared to all other methods regardless of the number of labeled training samples. On the other hand, the Active Transfer method on the IAM dataset performed comparable or slightly worse when judged against the Active method. This is likely due to the fact that the pre-trained RIMES model weights were not as helpful to the IAM dataset which is clearly seen when comparing the Scratch and Transfer methods. Thus, the weights initialization plays a large role in determining performance. Section 4.3 shows that the proposed evaluation metrics can predict how a set of model weights will perform in this framework.

These results indicate a substantial reduction in the amount of labeled samples that are necessary to achieve performance comparable to that of a model trained using the standard Scratch method using all available data. For the RIMES dataset, the Scratch method achieved a character error rate of 5.20% using 49,000 labeled samples. To achieve the same character error rate using the Active Transfer method, only 18,000 labeled samples are required. If we estimate it takes 15 seconds to label each image, we can assume it takes roughly 204 hours to annotate 49,000 samples. Reducing the required samples to 18,000 requires a much lower annotation effort of 75 hours, which is approximately a 63% reduction.

![Fig. 3. Results of the incremental training method compared to active and scratch training on the RIMES dataset. Actively selecting training samples obviously provides a significant benefit when using fewer labeled samples. The Incremental Scratch method seems to be fairly comparable with the Scratch method throughout most of the training process. When all available data is used, the method achieves similar performance to Active Scratch. In this example, incremental training alone usually does not produce better results, but the combination of incremental training with active sampling can provide higher model performance.](image-url)
We provide the full results of all methods for the RIMES, IAM, and CVL datasets in Table 1. Although labeling reductions are significant, the reader should keep in mind that such reductions may not be possible when the source dataset for the pre-trained weights initialization is a substantially different style compared to the target dataset (i.e. handwriting style, parchment color, size of text, stroke thickness, etc.). An example of this is with the IAM dataset in Table 1, where little improvement is observed with or without the pre-trained weights initialization. With this in mind, the Active Transfer method seems to be an effective method for training handwriting recognition models, especially when the right pre-trained model is used to initialize weights.

It’s also worthy to note the interesting results of the CVL dataset in Table 1. Using the pre-trained IAM model as the weights initialization was extremely beneficial as the Transfer method achieved similar performance to Active Transfer with the character error rate. However, the Active Transfer method was clearly superior to Transfer in reducing the word error rate. This is likely due to the fact that our uncertainty function for actively selecting samples provides an uncertainty for an entire word. Thus, words underrepresented in the dataset used for the pre-trained model weights will likely receive a greater uncertainty value and will be selected for labeling before other samples. Our uncertainty function inherently favors uncertain words, rather than uncertain characters. In the case of the CVL dataset in Table 1, the dramatic difference in word error rate between the Transfer and Active Transfer methods is likely due to the small amount of unique words available in the CVL dataset (See Appendix B). Thus, if we can actively sample for labeling those unique or commonly used words, we should be able to reduce our word error rate sooner.

### 4.2 Incremental Iterative Training

It is interesting to note that for all datasets in Figure 2, the Active and Active Transfer methods achieved better error rates compared to the Scratch and Transfer methods even when all labeled samples were used. Our results empirically show a benefit to an incremental training approach where $k$ samples are added to the training pool and the model is iteratively trained with the pool of labeled data and which size is increased by a value of $k$ each iteration.

To ensure that the Active and Active Transfer methods were not obtaining better results solely due to an incremental training approach, we incrementally trained a model from scratch with $k$ random samples (as opposed to $k$ actively selected samples) being added to the pool of labeled data each iteration. For this experiment, we again used a value of $k = 1000$ and incrementally trained this model on the RIMES dataset. Figure 3 provides the results of this method compared to the Active Scratch and Scratch methods. Our results show that there is an obvious benefit to actively selecting labeled samples using an uncertainty function as opposed to an incrementally trained model using random samples, especially when using fewer labeled samples. For most of the training process, the Scratch and Incremental Scratch methods achieve fairly comparable perfor-
Fig. 4. Comparison of the Active Transfer method on the RIMES dataset using three different source weight initializations: CVL, IAM, CVL + IAM. The set of weights used for initialization seems to have a fairly large impact on the overall performance of the Active Transfer method.

4.3 Selection of Pre-trained Model Weights

Figure 4 provides the results of the Active Transfer method on the RIMES dataset while varying the source of the pre-trained weights. We observe the superiority of the IAM dataset compared to the CVL and combination of IAM and CVL datasets.

To ensure the two evaluation methods previously described for selecting weights for initialization will be good indications of how the model will perform overall, we provide these metrics in Table 2. Empirical results show that both evaluation metrics seem to be good indicators of superior pre-trained model weights when incorporated in the active transfer framework. The IAM dataset was superior to the CVL dataset and the model selection evaluation metrics agreed. The combined CVL + IAM dataset seemed to achieve comparable performance to IAM alone. The final error rate for CVL + IAM was slightly better, but IAM alone took slightly fewer samples to achieve comparable Scratch performance.

5 Conclusion

In this work, we have proposed an active transfer learning solution for fitting handwriting recognition models to new datasets. Empirical results showed a
Model Weight Selection Metrics on the RIMES Dataset

<table>
<thead>
<tr>
<th>Source Weights</th>
<th>Confidence Eval</th>
<th>CER Eval</th>
<th>Samples(CER)</th>
<th>Final CER</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVL</td>
<td>25.04%</td>
<td>74.65%</td>
<td>30,000</td>
<td>4.84%</td>
</tr>
<tr>
<td>IAM</td>
<td>49.46%</td>
<td>36.30%</td>
<td>18,000</td>
<td>4.33%</td>
</tr>
<tr>
<td>CVL + IAM</td>
<td>42.29%</td>
<td>38.96%</td>
<td>20,000</td>
<td>4.24%</td>
</tr>
</tbody>
</table>

Table 2. Model weight evaluation metrics and final character error rates for the RIMES dataset using various pre-trained models for the weights initialization. Confidence Eval is calculated using Equation 1. CER Eval provides the character error rate on the target domain’s test set using the pre-trained model weights from the source domain. Both weight evaluation metrics are obtained before any fine-tuning is performed on the model. Samples(CER) is the minimum number of samples needed to reach the Scratch method’s best character error rate, while Final CER is the character error rate on the test set after training a model using the Active Transfer method. The best character error rate achieved on the RIMES dataset using all available training data was 5.20%.

significant reduction in the manual annotation effort required to achieve performance comparable to a fully trained model from scratch. This method was also shown to be superior to traditional active or transfer learning techniques, especially when a strong pre-trained model is used. We believe that our proposed method can be especially useful when a new collection of handwritten records requires recognition but contains a slightly different style or format. This could be applied to a task such as transcribing journal entries where handwriting is of a consistent style, but achieves mediocre results with a more generalized pre-trained model. By using this method, a substantially smaller amount of labeled samples may be required to achieve high-quality recognition.

The active transfer method proved to be effective even when using the simplest of transfer learning techniques. The proposed implementation only considers the use of pre-trained model weights for transfer learning with an incremental active learning training process. Future work should consider additional, more sophisticated transfer learning methods as described in Section 2 such as domain adversarial learning, synthetic data creation, and self-supervision to be inserted in the Active Transfer framework. In addition, the experiments conducted in this work only considered a few datasets that are fairly similar in style. Future work should also consider the implications when transferring model weights from a source domain that is substantially different from the target domain and when recognition is performed at the line and paragraph level rather than the word level.

References


A Data Augmentation Strategies

All of the following data augmentation strategies are used when pre-training a recognition model using SeqCLR [1]. The random grid warp augmentation is the only augmentation method used when training the model on labeled target data.

- **Random Noise**: This simple augmentation applies a random amount of noise to each pixel in the focus image.
- **Random Strokes**: Handwriting recognition is usually performed at the word or line-level and is often preceded by a segmentation step. These segmented word or line snippets will often contain stray ascenders or descenders from above or below lines. This augmentation is meant to simulate that effect. This is done by taking one or two random images, translating them up or down, and placing them on top of the focus image.
- **Bleedthrough**: This augmentation is meant to simulate the bleedthrough effect that occurs in historical documents, where ink from the flipped side of the page is visible. This is performed by taking one or two random images, mirroring them, and then placing the focus image on top.
- **Sharpen**: This is a simple sharpening augmentation. We randomly select the intensity from a uniform distribution between the interval $[0.1, 0.5]$. 
- **Blur**: This is a simple Gaussian blurring augmentation. We randomly select the intensity of $\sigma \in (0.5, 1.0)$. 
- **Vertical Crop**: Horizontal cropping will in most instances remove important information from the handwritten sequence. However, as long as the crop is not too severe, a vertical crop can be an effective sequence preserving augmentation. This idea was introduced in [1]. We uniformly select a percentage of the image to crop between the interval $[0.7, 0.9]$. 
- **Rotation**: The rotation augmentation cannot be too severe or else sequence information may be removed. Thus, we rotate the image uniformly between the interval $[-6^\circ, 6^\circ]$. 

– **Shear**: We perform a shear augmentation along the x-axis. To preserve sequence information, we randomly shear the image with a level selected uniformly between the interval \([-0.25, 0.25]\).

– **Cutout**: The cutout augmentation works by cutting out a small region from the focus image. We randomly select a height of \(h \in (4, 8)\) and a width of \(w \in (64, 128)\). The cutout of the selected size is then randomly positioned on the screen and filled randomly with either white or black pixels.

– **Random Grid Warp**: Random grid warp distortions are described extensively in [32] and model a significant amount of variability in handwriting. The augmentation works by placing an imaginary grid on top of an image and perturbing the grid intersections (called control points) with random noise taken from a Gaussian. For all of our experiments we use a grid interval of 16 and a standard deviation of 2.

### B  Dataset Descriptions

The following datasets are used in this work and a more detailed description of them are given here.

– **IAM Dataset**: The IAM dataset is a modern English dataset with more than 115,000 words and over 650 different writers. We choose to use the standard train/test split for the Large Writer Independent Recognition Task which contains just under 54,000 images for training and over 17,000 test images. The number of unique words in the training set is just over 7,100 [24].

– **RIMES Dataset**: The RIMES dataset is a modern French dataset with over 1,300 writers. The standard split contains over 51,000 training images and just shy of 8,000 test images. The number of unique words in the training set is just over 4,600 [16].

– **CVL Dataset**: CVL is a dataset containing modern handwriting with prompts originating from six English texts and one German. The limitation of this dataset is that the training set contains just over 250 unique words as the prompts originated from only seven texts [23].