Handwriting Recognition in Low-resource Scripts using Adversarial Learning

Ayan Kumar Bhunia[1] Abhirup Das[2] Ankan Kumar Bhunia[3] Perla Sai Raj Kishore[2] Partha Pratim Roy[4]
[1]Nanyang Technological University, Singapore [2] Institute of Engineering & Management, India
[3] Jadavpur University, India [4] Indian Institute of Technology Roorkee, India
[1]ayanbhunia007@gmail.com

Abstract

Handwritten Word Recognition and Spotting is a challenging field dealing with handwritten text possessing irregular and complex shapes. The design of deep neural network models makes it necessary to extend training datasets in order to introduce variations and increase the number of samples; word-retrieval is therefore very difficult in low-resource scripts. Much of the existing literature comprises preprocessing strategies which are seldom sufficient to cover all possible variations. We propose an Adversarial Feature Deformation Module (AFDM) that learns ways to elastically warp extracted features in a scalable manner. The AFDM is inserted between intermediate layers and trained alternatively with the original framework, boosting its capability to better learn highly informative features rather than trivial ones. We test our meta-framework, which is built on top of popular word-spotting and word-recognition frameworks and enhanced by AFDM, not only on extensive Latin word datasets but also on sparser Indic scripts. We record results for varying sizes of training data, and observe that our enhanced network generalizes much better in the low-data regime; the overall word-error rates and mAP scores are observed to improve as well.

1. Introduction

Handwriting recognition has been a very popular area of research over the last two decades, owing to handwritten documents being a personal choice of communication for humans, other than speech. The technology is applicable in postal automation, bank cheque processing, digitization of handwritten documents, and also as a reading aid for visually handicapped. Handwritten character recognition and word spotting and recognition systems have evolved significantly over the years. Since Nipkow's scanner [27] and LeNet [21], modern deep-learning based approaches today [18, 29, 41] seek to be able to robustly recognize handwritten text by learning local invariant patterns across diverse handwriting styles that are consistent in individual characters and scripts. These deep learning algorithms require vast amounts of data to train models that are robust to real-world handwritten data. While large datasets of both word-level and separated handwritten characters are available for scripts like Latin, a large number of scripts with larger vocabularies have limited data, posing challenges in research in the areas of word-spotting and recognition in languages using these scripts.

Deep learning algorithms, which have emerged in recent times, enable networks to effectively extract informative features from inputs and automatically generate transcriptions [31] of images of handwritten text or spot [40] query words, with high accuracy. In the case of scripts where abundant training data is not available, Deep Neural Networks (DNNs) often fall short, overfitting on the training set and thus generalizing poorly during evaluation. Popular methods such as data augmentation allow models to use the existing data more effectively, while batch-normalization [15] and dropout [39] prevent overfitting. Augmentation strategies such as random translations, flips, rotations and addition of Gaussian noise to input samples are often used to extend the original dataset [20] and prove to be beneficial for not only limited but also large datasets like Imagenet [7]. The existing literature [6, 19, 29, 51] augment the training data prior to feature extraction before classifying over as many as 3755 character classes [51]. Such transformations, however, fail to incorporate the wide variations in writing style and the complex shapes assumed by characters in words, by virtue of the free-flowing nature of handwritten text. Due to the huge space of possible variances in handwritten images, training by generating deformed examples through such generic means is not sufficient as the network easily adapts to these policies. Models need to become robust to uncommon deformations in inputs by learning to effectively utilize the more informative invariances, and it is not optimal to utilize just "hard" examples to do so [34, 43]. Instead, we propose an adversarial-learning based framework for handwritten word retrieval tasks for low resource scripts in order to train deep networks from a limited number of samples.

Information retrieval from handwritten images can be mainly classified into two types: (a) Handwritten Word Recognition (HWR) which outputs the complete transcription of the word-image and (b) Handwritten Word Spotting (HWS) which finds occurrences of a query keyword (either a string or sample word-image) from a collection of sample word-images. The existing literature on deep-learning based word retrieval, which cover mostly English words, make use of large available datasets, or use image augmentation techniques to increase the number of training samples [19]. Bhunia *et al*. [3] proposed a cross-lingual framework for Indic scripts where training is performed using a script that is abundantly available and testing is done on the low-resource script using character-mapping. The feasibility of this approach mostly depends on the extent of similarity between source and target scripts. Antoniou *et al*. [2] proposed a data augmentation framework using Generative Adversarial Networks (GANs) which can generate augmented data for new classes in a one-shot setup.

Inspired by the recent success of adversarial learning for different tasks like cross-domain image translation [52], domain adaptation [44] etc. we propose a generative adversarial learning based paradigm to augment the word images in a high dimensional feature space using spatial transformations [17]. We term it as Adversarial Feature Deformation Module (AFDM) that is added on top of the original task network performing either recognition or spotting. It prevents the latter from overfitting to easily learnable and trivial features. Consequently, frameworks enhanced by the proposed module generalize well to real-world testing data with rare deformations. Both the adversarial generator (AFDM) and task network are trained jointly, where the adversarial generator intends to generate "hard" examples while the task network attempts to learn invariances to difficult variations, to gradually become better over time. In this paper, we make the following novel contributions:

1. We propose a scalable solution to HWR and HWS in low resource scripts using adversarial learning to augment the data in high-dimensional convolutional feature space. Various deformations introduced by the adversarial generator encourage the task network to learn from different variations of handwriting even from a limited amount of data.
2. We compare our adversarial augmentation method with different baselines, and it clearly shows that the proposed framework can improve the performance of state-of-the-art handwritten word spotting and recognition systems. Not only is the performance improved in the case of low-resource scripts, but models generalize better to real-world handwritten data as well.

2. Related Works

Handwriting recognition has been researched in great detail in the past and in-depth reviews exist about it [28]. Nevertheless, the search for a better and more accurate technique continues to date. Results presented in [16] show that models should preferably use word-embeddings over bag-of-n-grams approaches. Based on this, another approach [29] employed a ConvNet to estimate a frequency based profile of n-grams constituting spatial parts of the word in input images and correlated it with profiles of existing words in a dictionary, demonstrating an attribute-based word-encoding scheme. In [40], Sudholt *et al*. adopted the VGG-Net [37] and used the terminal fully connected layers to predict holistic representations of handwritten-words in images by embedding their pyramidal histogram of characters (PHOC [1]) attributes. Architectures such as [18, 40, 48] similarly embedded features into a textual embedding space. The paper [49] demonstrated a region-proposal network driven word-spotting mechanism, where the end-to-end model encodes regional features into a distributed word-embedding space, where searches are performed. Sequence discriminative training based on Connectionist Temporal Classification (CTC) criterion, proposed by Graves *et al*. in [10] for training RNNs [14] has attracted much attention and been widely used in works like [11, 31]. In Shi *et al*. [31], the sequence of image features engineered by the ConvNet is given to a recurrent network such as LSTM [11] or MDLSTM [45, 4] for computing word transcriptions. Authors in [19] additionally included an affine-transformation based attention mechanism to reorient original images spatially prior to sequence-to-sequence transcription for improved detection accuracy. In most of the aforementioned methods, it is important to preprocess images in different ways to extend the original dataset, as observed in [18, 19, 20, 29, 35].

The process of augmenting to extend datasets is seen even in the case of large extensive datasets [19, 7] and in works focusing on Chinese handwritten character recognition where there are close to 4000 classes in standard datasets. In a different class of approaches, the process of online hard example mining (OHEM) has proved effective, boosting accuracy in datasets by targeting the fewer "hard" examples in the dataset, as shown in [22, 34, 36, 46]. With the advent of adversarial learning and GANs in recent years, several approaches have incorporated generative modeling to create synthetic data that is realistic [8, 26, 50], following architectural guidelines described by Goodfellow *et al*. for stable GAN training [9]. Papers such as [2] use GANs to augment data in limited datasets by computing over a sample class image to output samples that belong to the same class.

A recent work by Wang *et al*. [47] describes an adversarial model that generates hard examples by using the generator [9] to incorporate occlusions as well as spatial deformations into the feature space, forcing the detector to adapt to uncommon and rare deformations in actual inputs

to the model. In our framework, we use a similar strategy to make our word-retrieval detector robust and invariant to all sorts of variations seen in natural images of handwritten text. Another similar approach, [38], also explores the use of adversarial learning in visual tracking and object detection and attempts to alleviate the class-imbalance problem in datasets, where it is observed that the amount of data in one class far exceeds another class. Having a larger number of easy to recognize samples in datasets deters the training process as the detector is unaware of more valuable "hard" examples.

3. Handwritten Word Retrieval Models

We use the CRNN [31] and PHOCNet [40] as the baseline framework for handwritten word recognition and spotting respectively; on top of these, we implement our adversarial augmentation method. Significantly, our model is a meta-framework in the sense that the augmentation module can be incorporated along with a ResNet-like architecture too, instead of the VGG-like architecture adopted originally in both frameworks.

Convolutional Recurrent Neural Network for HWR: Shi *et al*. [31] introduced an end-to-end trainable Convolutional Recurrent Neural Network with Connectionist Temporal Classification (CTC) loss which can handle word sequences of arbitrary length without character segmentation and can predict the transcription of out-of-vocabulary word images using both lexicon-based and lexicon free approaches. The 'Map-to-Sequence' layer [31] acts as the bridge between the convolutional and the recurrent layers. The input is first fed to the convolutional layers; a recurrent network is built to make a per-frame prediction for each frame of the extracted features. Finally, a transcription layer translates the prediction from the recurrent layers into a label sequence.

PHOCNet for HWS: The PHOCNet [40] is a state-of-the-art approach in word-spotting, achieving exemplary results for both QbE (Query by Example) and QbS (Query by String) methods. The model reduces images of handwritten words to encoded representations of their corresponding visual attributes. The PHOC label [40] of a word is obtained by segmenting it into histograms at multiple levels. The histograms of characters in a word and its n-grams are calculated and concatenated to obtain a final representation. Once trained, an estimated PHOC representation is predicted for input word-images of varying sizes, by using a Spatial Pyramid Pooling layer [13]. These semantic representations of query and word-images can be compared directly by simple nearest-neighbor search (for QbE) or compared with the output representation of the deep model with PHOC of word-images in the dataset (for QbS). The PHOCNet uses sigmoid activation to generate the histograms instead of Softmax, utilizing a multi-label classification approach.

4. Proposed Methodology

4.1. Overview

The generic augmentation techniques popularly observed in HWR and HWS frameworks are often insufficient for models to generalize to real-world handwritten data, especially in the case of low-resource scripts where existing datasets are small and cover only a fraction of irregularities observed in the real world. We propose a modular deformation network that is trained to learn a manifold of parameters seeking to deform the features learned by the original task network, thus encouraging it to adapt to difficult examples and uncommon irregularities.

Let T be the task network whose input is an image I. By task network, we mean either a word recognition [31] or word spotting network [40], and the corresponding task loss be L_{task} which can be either CTC loss [31] (for word recognition) or cross-entropy loss [40] (for word spotting); we will use the terms *task network* and *task loss* for simplicity of description. We introduce an Adversarial Feature Deformation Module (AFDM) after one of the intermediate layers of the task network. Let us consider the task network T to be dissected into three parts, namely T_A, T_B and R. R is the final label prediction part, predicting either the word-level transcription for recognition or PHOC labels [40] for word spotting; T_A and T_B are the two successive convolutional parts of task network T. The exact position of dissection between T_A and T_B is discussed in Section 5.1. Let us assume F is the output feature map of T_A, i.e. $\mathbf{F} = T_A(I)$. The warped feature-map, $\mathbf{F}'$, from AFDM is thereafter passed through T_B and R for final label prediction. While the complete task network T is trained with the objective to correctly predict the output label, the AFDM tries to deform the features so that T can not predict the correct label easily. T is thereby enforced to generalize better to more discriminative invariances and informative features in the handwritten text data. The feature deformation network $\mathcal{A}$ and task network T compete in this adversarial game while training. During inference, we only use T.

4.2. Adversarial Feature Deformation Module

The AFDM, inspired by Spatial Transformation Networks (STN) [17], fulfills our objective of warping the features learned by T_A to make recognition (or spotting) difficult for the task-network. The module uses its adversarial *Localisation Network* $\mathcal{A}$ to predict a set of parameters θ. These parameters are needed to compute the transformation matrix $\mathcal{T}_\theta$. The *Grid Generator* generates a sampling grid S by performing the transformation $\mathcal{T}_\theta$ on points in the grid S' representing coordinates in $\mathbf{F}'$. The obtained grid S repre-

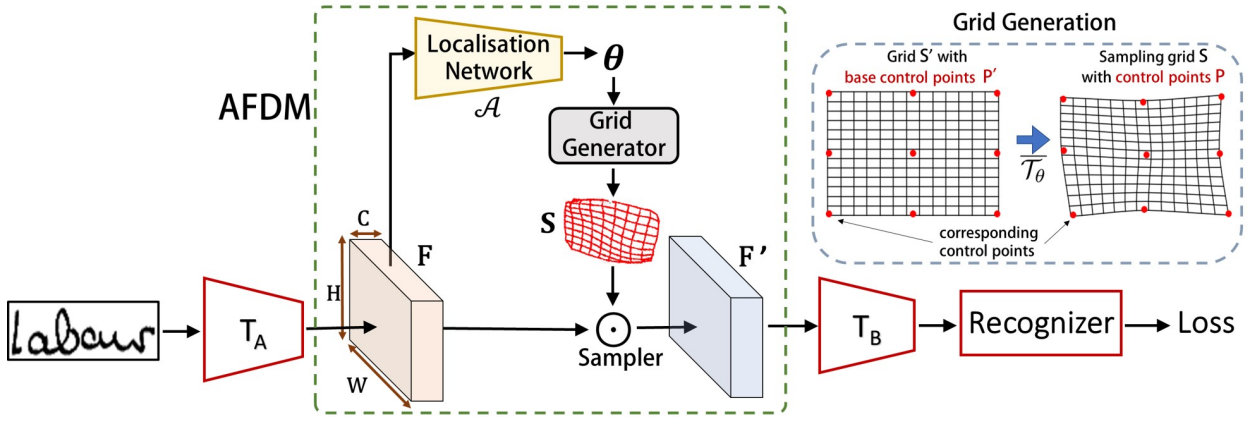

Figure 1: The architecture of our training network with the Adversarial Feature Deformation Module including the Localisation Network, Grid Generator and the Sampler inserted in between T_A and T_B of the task-network. The illustration depicts the use of the AFDM to uniformly deform the complete feature map $\mathbf{F}$.

sents N points in the original map $\mathbf{F}$ where corresponding points in the target map $\mathbf{F}'$ should be sampled from such that the latter appears spatially warped in the manner described by $\mathcal{T}_\theta$. This grid S and the original feature map are then passed through a *Bilinear Sampler* to obtain the target feature map $\mathbf{F}'$. Although a number of transformations [17] can be used in the AFDM, Thin Plate Spline Transformation (TPS) [5] is suggested to be the most powerful according to Jaderberg *et al*. [17]. We use TPS because of its degree of flexibility and ability to elastically deform a plane by solving a two-dimensional interpolation problem: the computation of a map $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ from a set of arbitrary control points [5]. Furthermore, the matrix operations for grid-generation and transformation in TPS being differentiable, the module can backpropagate gradients as well.

The parameters predicted adversarially by $\mathcal{A}$ denote K control points $P = [p_1, \cdots, p_K] \in \mathbb{R}^{2\times K}$ with $p_v = [x_v, y_v]^T$ pointing to coordinates in $\mathbf{F}$ by regressing over their x, y values, which are normalised to lie within [-1,1]. The Grid Generator uses the parameters representing the control points in P to define a transformation function for a set of corresponding control points $P' = [p'_1, \cdots, p'_K]$, called the *base control points* representing positions in $\mathbf{F}'$. Since the base control points are fixed, P' is a constant. The transformation is denoted by a matrix $\mathcal{T}_\theta \in \mathbb{R}^{2\times(K+3)}$ that can be computed as:

$$\mathcal{T}_\theta = \left(\Delta_{P'}^{-1} \begin{bmatrix} P^T \\ 0^{3\times 2} \end{bmatrix}\right)^T \tag{1}$$

where $\Delta_{P'} \in \mathbb{R}^{(K+3)\times(K+3)}$ is also a constant. It is given by:

$$\Delta_{P'} = \begin{pmatrix} 1^{K\times 1} & P'^T & \mathcal{E} \\ 0 & 0 & 1^{1\times K} \\ 0 & 0 & P' \end{pmatrix} \tag{2}$$

where, the element in i-th row and j-th column of matrix $\mathcal{E}$ represents the euclidean distance between the base control points p'_i and p'_j. Now, given that the grid of target points in $\mathbf{F}'$ is denoted as $S' = \{s'_i\}_{i=1,\cdots,N}$, with $s'_i = [\mathrm{x}'_i, \mathrm{y}'_i]^T$ being the x,y coordinates for the i-th point of a total of N feature-points, for every point s'_i we find the corresponding sampling position $s_i = [\mathrm{x}_i, \mathrm{y}_i]^T$ in $\mathbf{F}$ through the following steps:

$$e'_{i,k} = d^2_{i,k} \ln d^2_{i,k} \tag{3}$$

$$\hat{\mathrm{s}}'_i = [1, x'_i, y'_i, e'_{i,1}, \cdots, e'_{i,K}]^T \in \mathbb{R}^{(K+3)\times 1} \tag{4}$$

$$\mathrm{s}_i = \mathcal{T}_\theta \cdot \hat{\mathrm{s}}'_i \tag{5}$$

where $d_{i,k}$ is the euclidean distance between s'_i and k-th base control point p'_k. We iteratively modify all N points in S' using eqn. (5) to define the grid-transform function $\overline{\mathcal{T}_\theta}(\cdot)$ and produce sampling grid S:

$$S = \overline{\mathcal{T}_\theta}(\{s'_i\}), \quad i = 1, 2, \cdots, N \tag{6}$$

We obtain the grid $S = \{s_i\}_{i=1,\cdots,N}$ representing sampling points in $\mathbf{F}$.

The network represented by $\mathcal{A}$ includes a final fully-connected (fc) layer predicting 2K normalized coordinate values. It is fitted with the $\tanh(\cdot)$ activation function, after which the values are reshaped to form matrix the P. It is to be noted that the aforementioned equations define the deformation operation executed by the AFDM such that all channels in our original map are deformed uniformly. In the later sections, we discuss the partitioning strategy where smaller sub-maps in $\mathbf{F}$ are fed individually into it for deformation.

4.3. Adversarial Learning

Traditional approaches to adversarial learning [9] involve training a model to learn a generator G which given

a vector z sampled from a noise distribution $P_{noise}(z)$ outputs an image $G(z)$. The discriminator D takes either the generated image or real image x from distribution $P_{data}(x)$ as input, and identifies whether it is real or fake. The objective function for training the network using cross-entropy loss is defined as:

$$L = \min_G \max_D \mathbb{E}_{x\sim P_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim P_{noise}(z)}[log(1 - D(G(z)))] \quad (7)$$

In traditional GANs, Generator G learns a mapping of z from the noise distribution $P_{noise}(z)$ to the data distribution $P_{data}(x)$ over data x. In our framework, G (i.e. AFDM) learns a mapping of $\mathbf{F}$ from the distribution of *undistorted* features $P_{undistorted}(\mathbf{F})$ to the space of *distorted* features $P_{distorted}(\mathbf{F}')$.

$$L = \min_G \max_D \mathbb{E}_{\mathbf{F}'\sim P_{distorted}(\mathbf{F}')}[logD(\mathbf{F}')] + \mathbb{E}_{\mathbf{F}\sim P_{undistorted}(\mathbf{F})}[log(1 - D(G(\mathbf{F})))] \quad (8)$$

In our framework, we train AFDM $\mathcal{A}$ (analogous to G) and task network T (analogous to D) alternatively in an adversarial manner. Initially, $\mathcal{A}$ generates random deformations, but with the progress of adversarial learning, it learns strategies to warp the intermediate feature space so that it becomes hard to recognize (spot) for T. In other words, the the generator framework $\mathcal{A}$ tries to deform (see Figure 3) the feature map to make the task harder. Also, we seek to train the discriminating network, i.e. the task-network T in a supervised manner using labelled samples, while encouraging it to accurately retrieve handwritten inputs despite deformations present in them.

Now, instead of deforming $\mathbf{F}$ (with height H, width W and C channels) uniformly, we modify k sub-maps constituting it in k different ways (k being a value much smaller than the number of channels C), thereby increasing the complexity of the task and preventing $\mathcal{A}$ from learning trivial warping strategies. $\mathbf{F}$ is divided into sub-maps f_1 through f_k, each of which has $\frac{C}{k}$ channels. The m-th sub-map $f_m \in \mathbb{R}^{H\times W\times \frac{C}{k}}$ is then fed into the $\mathcal{A}$ to generate θ_m, and compute the grid-transform function $\overline{\mathcal{T}_{\theta_m}}(\cdot)$. The latter, as shown in eqn. 1 through 6, transforms a given grid S'_m to obtain the corresponding grid of sampling points S_m for points belonging to sub-map f_m. The deformed feature map $\mathbf{F}'$ is thus computed as:

$$\mathbf{F}' = (f_1 \odot S_1) \oplus (f_2 \odot S_2) \oplus \cdots \oplus (f_k \odot S_k) \quad (9)$$

where $\oplus$ denotes the channel-wise concatenation operation and $\odot$ denotes the bilinear-sampling mechanism corresponding to the transforms described in [17]. The sub-map f_m is thus sampled to obtain $(f_m \odot S_m) \in \mathbb{R}^{H\times W\times \frac{C}{k}}$, and concatenated to get $\mathbf{F}'$ having same dimensions as the original feature-map $\mathbf{F}$. The AFDM thus learns a function $\mathcal{A}(\cdot)$ that computes the encoded features in the m-th sub-map to generate $\theta_m = \mathcal{A}(f_m)$.

In absence of the AFDM (e.g. during testing), the output $\mathbf{F}$ of sub-network T_A is further passed through T_B and R. The recognizer R outputs the predicted word-label L_p for a word image I. The word-label can be either word-level annotation represented by series of character, or PHOC label [40] based on the type of system. Let us assume the ground-truth label for the latter be L_g. Thus our original word-retrieval loss L_{task} can be defined as:

$$L_{task} = Q_{word}(\mathrm{L}_p, \mathrm{L}_g) \quad (10)$$

where $Q_{word}(\cdot)$ represents a general function that computes loss between the prediction L_p and the ground truth label L_g, which is either the CTC loss used in [31] or the sigmoid-cross-entropy loss described in [40].

During training, we have two different networks: the task network T and Localisation Network $\mathcal{A}$. Let us consider their parameters to be θ_T and θ_A respectively. In one iteration during training, the data flow in the forward pass is as follows: $I \rightarrow T_A(\cdot) \rightarrow AFDM(\cdot) \rightarrow T_B(\cdot) \rightarrow R(\cdot) \rightarrow \mathrm{L}_p$, where $AFDM(\cdot)$ represents the complete deformation operation including parameter prediction by $\mathcal{A}$, grid-generation and sampling operations; the last two do not involve learning any parameters. $\mathcal{A}$ needs to learn feature deforming strategies through θ_A so that the recognizer should fail. We thus obtain θ_A by maximizing the loss function L_{task}. On the other hand, the θ_T is optimized to minimize the task loss L_{task}.

$$\theta_A = \arg\max_{\theta_A} L_{task} \quad (11)$$

$$\theta_T = \arg\min_{\theta_T} L_{task} \quad (12)$$

As a result, if the deformation caused by the AFDM makes image I hard to recognize, the task network T gets a high loss and $\mathcal{A}$ gets a low loss, else if the modified features are easy to recognize, the $\mathcal{A}$ suffers a high loss instead.

5. Experimentation Details

5.1. Datasets

We use two very popular datasets of Latin scripts, namely **IAM** (1,15,320 words) and **RIMES** (66,982 words) datasets, used by handwritten document image analysis community extensively. **IAM** [24] is one of the largest datasets available for HWR and HWS in Latin script, allowing us to demonstrate the effectiveness of our feature warping strategy at different sizes of training sets (see Figure 2). In order to demonstrate the effectiveness of our model in low-resource scripts (in terms of availability of training data), we choose two Indic scripts, namely Bangla and Devanagari (Hindi), as examples to demonstrate the benefits

	IAM		RIMES		IndicBAN		IndicDEV	
	WER	CER	WER	CER	WER	CER	WER	CER
Handwritten Word Recognition (Unconstrained)								
B1	23.14	12.02	16.04	11.17	26.31	14.67	25.35	13.69
B2	-	-	-	-	25.17	13.08	24.37	12.14
B3	21.58	11.45	14.61	10.37	20.28	11.13	19.07	10.34
B4	19.97	9.81	12.42	7.61	17.67	9.19	16.46	8.34
RARE[32]	19.72	9.69	12.32	7.65	20.15	10.52	19.19	9.72
ASTER[33]	**17.01**	**8.11**	10.52	6.62	18.31	9.22	17.22	8.13
MORAN[23]	17.95	8.96	11.27	7.05	19.62	9.83	18.56	9.01
Ours	17.19	8.41	**10.47**	**6.44**	**15.47**	**7.12**	**14.3**	**6.14**
Handwritten Word Recognition (Lexicon)								
B1	15.98	10.05	12.51	9.64	16.67	10.21	15.67	9.78
B2	-	-	-	-	15.87	9.47	14.69	8.41
B3	12.17	8.45	10.13	7.17	11.37	7.64	10.24	6.76
B4	10.24	7.21	7.59	5.56	9.69	5.41	8.67	4.67
RARE[32]	9.93	7.33	7.48	5.25	11.13	7.55	10.20	6.68
ASTER[33]	**8.73**	**5.83**	**6.17**	**3.12**	9.44	5.42	8.61	4.59
MORAN[23]	9.19	6.52	6.83	3.83	10.52	6.61	9.68	5.46
Ours	8.87	5.94	6.31	3.17	**7.49**	**4.37**	**6.59**	**3.97**
Handwritten Word Spotting								
	QbS	QbE	QbS	QbE	QbS	QbE	QbS	QbE
B1	83.12	72.67	86.31	77.69	80.37	76.91	81.67	77.61
B2	-	-	-	-	81.04	77.67	82.64	78.64
B3	85.1	73.67	87.69	79.67	84.67	84.73	85.61	86.19
B4	86.94	75.64	90.34	80.67	87.67	85.49	88.17	86.49
TPP-PHOCNet[42]	**92.97**	**84.80**	**94.31**	**85.89**	89.21	**86.69**	89.97	**87.82**
Ours	88.69	77.94	92.94	82.67	**89.34**	86.47	**90.13**	87.67

Table 1: Performance in (a) Handwritten Word Recognition (HWR) and (b) Handwritten Word Spotting (HWS) of our baselines as well as state-of-the-Art approaches on different datasets.

of adversarial training via AFDM. Hindi and Bangla are the fifth and sixth most popular languages globally [27] and use the scripts Devanagari and Bangla, respectively. Both scripts are far more complex than Latin due to the presence of modifiers[30] and complex cursive shapes [30] and are sparse compared to Latin [12, 24]. To the best of our knowledge, there exists only one publicly available dataset [3, 30] which contains a total of 17,091 and 16,128 words for Bangla and Devanagari, respectively. We denote these two datasets as **IndBAN** (BANgla) and **IndDEV** (DEVanagari) respectively. For IAM, IndBAN and IndDEV, we use the same partition for training, validation and testing provided along with the datasets. For RIMES dataset, we follow the partition released by ICDAR 2011 competition.

5.2. Implementation Details

While experimenting, we notice that it is important to first pre-train the task network for a certain number of iterations so that it can learn a basic model to understand the shapes of different characters to an extent. If we start training both the networks together, we notice that the AFDM often overpowers the task network and it fails to learn meaningful representation. Therefore, we first train the task network for 10K iterations without the AFDM. Thereafter, we include the latter to fulfill its adversarial objective of deforming the intermediate convolutional feature maps. We use 500 continuous iterations to train the parameter localization network $\mathcal{A}$ alone for better initialization. It is observed that due to the large degree of flexibility TPS often finds some especially difficult deformations which task network fails to generalize later on. Hence, we use a simple trick to solve this stability issue: we only deform half of the data samples randomly in a batch through the AFDM and the rest are kept unchanged for retrieval; this greatly improves the stability issue. For the *Localisation Network*, we use four convolutional layers with stride 2 and kernel size 3×3 followed by 2 fully-connected layers, finally predicting 18 parameter values using $\tanh$ activation. We keep the number of sub-map divisions (k) to 4. We use a batch size of 32. Following the earlier initialization, both the task-network and AFDM are trained for a total of 100K iterations alternatively. We use Adam optimizer for both task network and AFDM, however, we keep the learning rate for task network to 10^{-4} and the same for the Localisation Network of AFDM is 10^{-3}. PHOCNet consists of 13 convolutional layers followed by an SPP layer and 3 fully connected layers and finally predicting the PHOC labels using sigmoid activation. We name these conv-layers as follows: conv1_1, conv1_2, conv2_1, conv2_2, conv3_1, conv3_2, conv3_3, conv3_4, conv3_5, conv3_6, conv4_1, conv4_2 and conv4_3. There are two pooling layers (2×2) after conv1_2 and conv2_2. Every convolution layer has a kernel of size 3×3 and number of filters are 64, 128, 256, and 512 for conv1_X, conv2_X, conv3_X, conv4_X respectively. On the other hand, our CRNN framework comprises of 8 conv layers, followed by a 'Map-to-Sequence' and a 2-layer BLSTM unit. The architecture is: conv1, conv2, conv3_1, conv3_2, conv4_1, conv4_2, conv5_1, conv5_2, conv6. The first 7 layers have 3×3 kernels but the last layer has a 2×2 kernel. There are 64, 128 and 256 filters in conv1_X, conv2_X, conv3_X, and 512 filters from conv4_1 till conv6; the pooling layers are after conv1, conv2, conv3_2, conv4_2 and conv5_2. While the pooling windows of the first two pool layers are 2×2, the rest are 1×2. Based on the experimental analysis, we introduce AFDM after conv4_1 layer in PHOCNet, and after conv4_1 layer in CRNN. It is to be noted that the input is resized to a height of 64 keeping the aspect ratio same. More analysis is given in Section 5.5.

5.3. Baseline Methods

To the best of our knowledge, there is no prior work dealing with adversarial data augmentation strategy for HWS and HWR. Based on different popular data augmentation and transfer learning strategies, we define a couple of baselines to demonstrate the effectiveness of the AFDM.

- **B1**: In this baseline, we perform different image-level

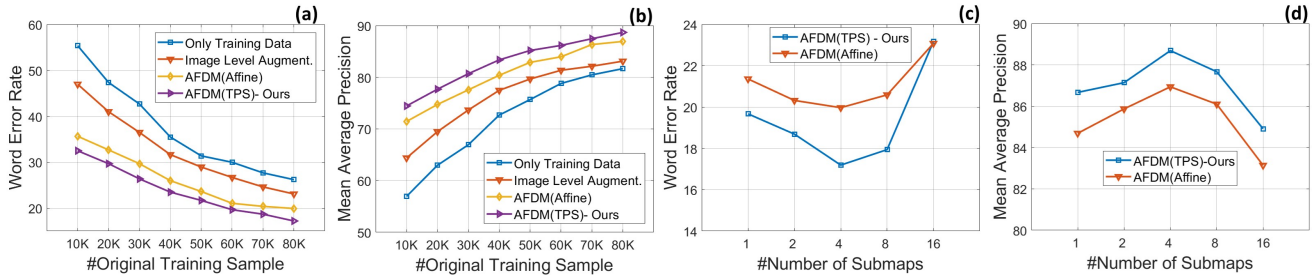

Figure 2: (a) Word Error Rate (WER) for HWR (unconstrained) and (b) mean Average Precision (mAP) for QbS in HWS for different number of training samples on standard testing set using different data augmentation strategies on IAM dataset. (c) and (d) represent the performance using different sub-map partitioning schemes; the setup is described in Section 5.2.

Layers	conv3_1	conv3_2	conv3_3	conv3_4	conv3_5	conv3_6	conv4_1	conv4_2	conv4_3
AFDM(TPS)	85.29	85.20	85.97	86.94	87.88	88.19	**88.69**	87.81	87.77
AFDM(Affine)	84.13	84.12	84.53	85.01	85.33	85.81	**86.94**	86.02	85.24

Table 2: Mean Average Precision (mAP) on using AFDM after a specific layer in PHOCNet for Query by String.

Layers	conv3_1	conv3_2	conv4_1	conv4_2	conv5_1	conv5_1
A-TPS	17.98	17.41	**17.19**	17.25	20.32	20.41
A-Affine	22.01	20.11	**19.97**	20.01	19.99	20.21

Table 3: Word Error Rate(WER) on using AFDM after a specific layer in CRNN (unconstrained).

data augmentation techniques mentioned in [29] and [40] on the handwritten word images to increase the total number of word samples ($\sim 500K$) in the training set.

• **B2**: Here we use transfer learning strategy to alleviate the problem of data insufficiency in low resource scripts. We train both HWR and HWS model using a large amount of data present in Latin scripts, thereafter we fix the weights till conv5_2 (conv4_2) layer of the CRNN (PHOCNet) network and we fine-tune rest of the layers over the available annotated data from Indic scripts.

• **B3**: This is identical to our adversarial learning based framework, except that it deforms data in the image-space using the TPS mechanism (Section 4.2). The input to the AFDM is the original training image.

• **B4**: Here, we use affine transformation [17] in place of TPS, using a fewer number of parameters (six) to devise warping policies with relatively less degree of freedom for deformation.

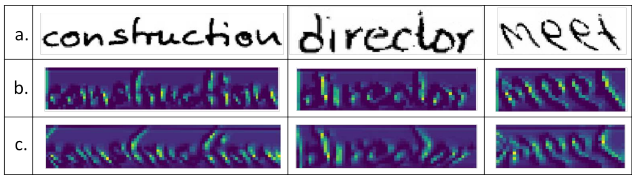

Figure 3: We show visualization with three examples (column-wise): the original image (a) and the first channel of the *undistorted* feature map (b) as well as the *distorted* feature map (c).

5.4. Performance on HWR and HWS

In our experiments, we use Character Error Rate (CER) and Word Error Rate (WER) as metrics [4] for HWR, while mean Average Precision (mAP) metric [40] is considered for HWS. In case of lexicon based recognition for IAM dataset, we use all the unique words present in the dataset, whereas we use lexicon provided in ICDAR 2011 competition for RIMES dataset and the lexicons provided with the original dataset are used for IndBAN and IndDEV datasets. From Table 1, it is to be noted that our adversarial feature augmentation method using TPS significantly outperforms **B1** which uses different image level data augmentation techniques as seen in [29, 31] together. This signifies that only image level "handcrafted" data augmentation cannot improve the performance significantly even if we increase the the number of data-samples through possible transformations. We notice that weight initialization from pretrained weights in **B2** helps to increase the performance for both HWR and HWS to a reasonable extent and also speeds up the training procedure significantly. Both **B3** and **B4** are adversarial frameworks. From the results on both HWR and HWS, it can be concluded that adversarial data augmentation works better while introduced in the intermediate layers of the convolutional network rather than adversarial deformation in image space as done in **B3**. Also, TPS performs better than simple affine transformation in **B4** due to greater degree of flexibility in deformation.

Overall, the improvement due to adversarial data augmentation is clearly higher for both IndBAN and IndDEV. Also, performance is better in IndBAN and IndDEV dataset than other two datasets inspite of our claim of having more complexity in Bangla and Devanagari script. The major rea-

son behind this is that IndBAN and IndDEV datasets have multiple copies of same words by same author in both training and testing sets as well as simpler words (having 4 characters on average), while the IAM dataset has more complex samples in testing sets. Word retrieval in real-world scenarios of Bangla and Devanagari script is far more complex than what it is in unseen testing set. Moreover, due to limited training data as well as a large number of character classes [3, 30], image level data augmentation cannot generalize the model well in testing set, giving poor performance for both HWR and HWS. In contrast, the proposed method using adversarial learning helps in significant performance gain compared to image level data augmentation. We have also compared the recent state-of-the-art methods [33, 42, 23, 32] with ours. Note that it is a meta framework; the AFDM module can be incorporated in [33, 42, 23, 32] too. Overall, our results are competitive with recent frameworks on popular datasets like IAM and RIMES and show reasonable improvement on low resource scripts (e.g. IndicBAN and IndicDEV).

5.5. Ablation Study

We have comprehensively studied the improvements achieved from different augmentation techniques at various training data sizes on the IAM dataset. We experiment over 8 instances with our training set size ranging from 10K to 80K, using the standard testing set for evaluation. From Figure 2, it is evident that the proposed method performs well in the low-data regime, producing a reasonable improvement over image-level augmentation. It is to be noted that with increasing training data, the improvement gained by our model over other baselines (which do not use adversarial augmentation) gets reduced. We also evaluated the performance by including the AFDM at different positions of the ConvNet feature extraction units in PHOCNet and CRNN (shown in Table 2 and 3). We observe that if the AFDM is inserted between shallower layers, the model diverges and we do not achieve a desirable result. Better performance along with improved stability in training is observed in the mid-to-deeper parts of the task network which encode a higher-level understanding of the extracted feature information. The performance again drops at very deep layers. We also evaluate the performance of the model by partitioning the original feature map into 1, 2, 4, 8 and 16 sub-maps using rest of the standard setup. It was noticed that 4 divisions provide the optimum result (Figure 2).

Adversarial *vs*. Non-adversarial Learning: In contrast to AFDM that is based on STN [17] and trained using adversarial objective, an alternative (*non-adversarial*) could be the work by Shi *et al*. [32] where STN is used to rectify the spatial orientation of a word-image to make recognition easier for Sequence Recognition Network [32] according to the original philosophy of [17]. Following [32], we introduce an STN module with TPS before the CRNN and PHOCNet architecture and train the complete architecture (STN + Task Network) in an end-to-end manner with the task loss objective (Equation 10), keeping rest of the standard experimental setup of the task network same. The unconstrained WER for non-adversarial pipeline using STN is 20.07% and the mAP value for QbS is 85.64%, trailing behind the proposed adversarial framework by 3.51% (WER) and 3.05% (mAP) respectively. Next, we divide the IAM dataset into hard and easy word samples using the framework of Mor *et al*. [25] with CRNN as baseline recognizer. We consider top 70% word images as easy samples and 30% as hard samples based on the confidence score. High score signifies easily recognizable images without much deformation, while images with lower scores contain ample deformation in them. We train both the adversarial and non-adversarial pipeline using these easy samples and test on hard samples. This experimental setup challenges the models to learn invariance that can generalize for hard unseen word samples which are absent during training. It is observed that while non-adversarial pipeline provides 40.22% unconstrained WER (71.31 mAP-QbS), our adversarial framework achieves 27.64% WER (82.67 mAP-QbS) outperforming the non-adversarial alternatives by a large margin of 12.58% WER (11.36 mAP-QbS). Although the objective of both of these pipelines is to learn a robust model invariant to different types of deformation in handwritten data, the non-adversarial method tries to learn the invariance only from available training data while failing to generalize on unseen irregularities and deformations. Due to free-flow nature of handwriting, it is not possible to include every possible variation in the training dataset. Hence, our adversarial framework proves useful to learn a robust model that can generalize well on unseen deformations which are absent in sparse datasets.

6. Conclusion

We study a common difficulty often faced by researchers exploring handwriting recognition in low-resource scripts and try to overcome the limitations of generic data augmentation strategies. The AFDM can be flexibly added to frameworks for both word-spotting and recognition, allowing deep networks to generalize well even in low-data settings. Rather than augmenting handwritten data in image space using "handcrafted" techniques, adversarially warping the intermediate feature-space using TPS is a scalable solution to overcome the dearth of variations seen in some sparse training datasets. The higher degree of flexibility incorporated by TPS with the adversarial parameterisation strategy goes a long way to incorporate rare unseen variations, beating deformation policies that frameworks can easily overfit to.

References

[1] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny. Word spotting and recognition with embedded attributes. *IEEE transactions on pattern analysis and machine intelligence*, 36(12):2552–2566, 2014. 2

[2] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017. 2

[3] Ayan Kumar Bhunia, Partha Pratim Roy, Akash Mohta, and Umapada Pal. Cross-language framework for word recognition and spotting of indic scripts. *Pattern Recognition*, 79:12–31, 2018. 2, 6, 8

[4] Théodore Bluche, Jérôome Louradour, and Ronaldo Messina. Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention. In *ICDAR*, volume 1, pages 1050–1055, 2017. 2, 7

[5] Fred L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence*, 11(6):567–585, 1989. 4

[6] Dan Cireşan and Ueli Meier. Multi-column deep neural networks for offline handwritten chinese character classification. In *IJCNN*, pages 1–6, 2015. 1

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 1, 2

[8] Hao Dong, Paarth Neekhara, Chao Wu, and Yike Guo. Unsupervised image-to-image translation with generative adversarial networks. *arXiv preprint arXiv:1701.02676*, 2017. 2

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2, 4

[10] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, pages 369–376, 2006. 2

[11] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868, 2009. 2

[12] Emmanuele Grosicki and Haikal El-Abed. Icdar 2011-french handwriting recognition competition. In *ICDAR*, pages 1459–1463, 2011. 6

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, pages 346–361, 2014. 3

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37, pages 448–456, 2015. 1

[16] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014. 2

[17] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 2, 3, 4, 5, 7, 8

[18] Praveen Krishnan, Kartik Dutta, and CV Jawahar. Deep feature embedding for accurate recognition and retrieval of handwritten text. In *ICFHR*, pages 289–294, 2016. 1, 2

[19] Praveen Krishnan, Kartik Dutta, and CV Jawahar. Word spotting and recognition using deep embedding. In *DAS*, pages 1–6, 2018. 1, 2

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1, 2

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[22] Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015. 2

[23] Canjie Luo, Lianwen Jin, and Zenghui Sun. Moran: A multi-object rectified attention network for scene text recognition. *Pattern Recognition*, 2019. 6, 8

[24] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *IJDAR*, 5(1):39–46, 2002. 5, 6

[25] Noam Mor and Lior Wolf. Confidence prediction for lexicon-free ocr. In *WACV*, pages 218–225, 2018. 8

[26] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016. 2

[27] U Pal and BB Chaudhuri. Indian script character recognition: a survey. *Pattern Recognition*, 37(9):1887–1899, 2004. 1, 6

[28] Réjean Plamondon and Sargur N Srihari. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):63–84, 2000. 2

[29] Arik Poznanski and Lior Wolf. Cnn-n-gram for handwriting word recognition. In *CVPR*, pages 2305–2314, 2016. 1, 2, 7

[30] Partha Pratim Roy, Ayan Kumar Bhunia, Ayan Das, Prasenjit Dey, and Umapada Pal. Hmm-based indic handwritten word recognition using zone segmentation. *Pattern Recognition*, 60:1057–1075, 2016. 6, 8

[31] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2017. 1, 2, 3, 5, 7

[32] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. In *CVPR*, pages 4168–4176, 2016. 6, 8

[33] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 6, 8

[34] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, pages 761–769, 2016. 1, 2

[35] Patrice Y Simard, Dave Steinkraus, and John C Platt. Best practices for convolutional neural networks applied to visual document analysis. In *ICDAR*, page 958, 2003. 2

[36] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, and Francesc Moreno-Noguer. Fracking deep convolutional image descriptors. *arXiv preprint arXiv:1412.6537*, 2014. 2

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[38] Yibing Song, Chao Ma, Xiaohe Wu, Lijun Gong, Linchao Bao, Wangmeng Zuo, Chunhua Shen, Rynson Lau, and Ming-Hsuan Yang. Vital: Visual tracking via adversarial learning. *arXiv preprint arXiv:1804.04273*, 2018. 3

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 1

[40] Sebastian Sudholt and Gernot A Fink. Phocnet: A deep convolutional neural network for word spotting in handwritten documents. In *ICFHR*, pages 277–282, 2016. 1, 2, 3, 5, 7

[41] Sebastian Sudholt and Gernot A Fink. Evaluating word string embeddings and loss functions for cnn-based word spotting. In *ICDAR*, volume 1, pages 493–498, 2017. 1

[42] Sebastian Sudholt and Gernot A Fink. Attribute cnns for word spotting in handwritten documents. *International Journal on Document Analysis and Recognition (IJDAR)*, 21(3):199–218, 2018. 6, 8

[43] Martin Takác, Avleen Singh Bijral, Peter Richtárik, and Nati Srebro. Mini-batch primal and dual methods for svms. In *ICML (3)*, pages 1022–1030, 2013. 1

[44] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, volume 1, page 4, 2017. 2

[45] Paul Voigtlaender, Patrick Doetsch, and Hermann Ney. Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In *ICFHR*, pages 228–233, 2016. 2

[46] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, pages 2794–2802, 2015. 2

[47] Xiaolong Wang, Abhinav Shrivastava, and Abhinav Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *CVPR*, 2017. 2

[48] Tomas Wilkinson and Anders Brun. Semantic and verbatim word spotting using deep neural networks. In *ICFHR*, pages 307–312, 2016. 2

[49] Tomas Wilkinson, ICCVJonas Lindström, and Anders Brun. Neural ctrl-f: segmentation-free query-by-string word spotting in handwritten manuscript collections. In *ICCV*, pages 4443–4452, 2017. 2

[50] Raymond Yeh, Chen Chen, Teck Yian Lim, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with perceptual and contextual losses. arxiv preprint. *arXiv preprint arXiv:1607.07539*, 2, 2016. 2

[51] Zhuoyao Zhong, Lianwen Jin, and Zecheng Xie. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In *ICDAR*, pages 846–850, 2015. 1

[52] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017. 2