TextAdaIN: Paying Attention to Shortcut Learning in Text Recognizers Supplementary Material

Oren Nuriel, Sharon Fogel, and Ron Litman

AWS AI Labs
{onuriel, shafog, litmanr}@amazon.com

A SCATTER - Baseline Architecture

As mentioned in the paper, throughout this work our baseline architecture is a state-of-the-art recognizer named SCATTER [16]. The SCATTER architecture consists of four main components:

1. Transformation: A rectification module that aligns the input text image using a Thin Plate Spline (TPS) transformation [17, 23].
2. Feature extraction: A convolutional neural network (CNN) that extracts features from the rectified image. Similar to [4, 3, 16, 2, 1], a 29-layer ResNet is employed as the backbone. Subsequently, the features are mapped to a sequence of frames, denoted by $V = [v_1, v_2, ..., v_T]$, each corresponding to different receptive fields in the image.
3. Visual feature refinement: An intermediate supervision in the form of a CTC decoder [7] is employed to provide direct supervision for each frame in the visual features V.
4. Selective-Contextual Refinement Block: Contextual features are extracted using a two-layer bi-directional LSTM encoder. These features are concatenated to the visual features, V. The features are then fed into a selective decoder and into a subsequent block if it exists. These blocks can be stacked together to improve results. In this work, for convenience, we set the number of blocks to two.

B Image Corruptions

In Sections 3.1 and 5.4, we compare the performance of the baseline model and the TextAdaIN version on corrupted versions of the IAM test set. Fig. 1 contains original images from the test set on the first column and the results of applying each of the corruptions on the following columns. They are split into three categories based on their impact: (a) local masking, (b) pixel-wise distortions and (c) geometric. The corruptions are applied utilizing the *imgaug* [11] package as explicitly written below.

```
from imgaug import augmenters as iaa
corruptions = {
    'original': iaa.Noop(),
    'dropout': iaa.CoarseDropout((0.0, 0.05),
```

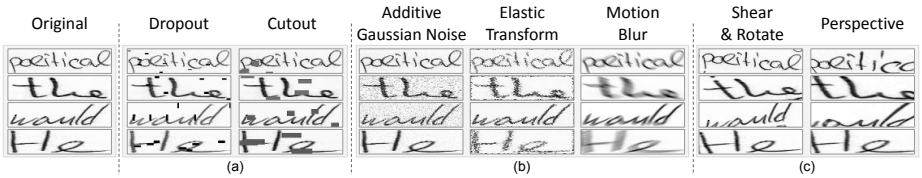

Fig. 1: **Visualization of selected corruptions**. To assert TextAdaIN's advantage on challenging testing conditions, we compare its performance to the baseline while applying different types of corruptions. We visualize the different corruptions and divide them into categories based on their impact: local masking, pixel-wise distortions and geometric.

```
    'cutout': iaa.Cutout(nb_iterations=4),
    'additiive_gaussian_noise': iaa.AdditiveGaussianNoise(scale
    =(0, 0.2*255)),
    'elastic_transformation':  iaa.ElasticTransformation(alpha
    =(0, 5.0), sigma=0.5),
    'motion_blur': iaa.MotionBlur(k=15, angle=[-45, 45]),
    'shear_rotate': iaa.Affine(shear=(-10, 10),rotate=(-10, 10),
     mode='reflect'),
 size_percent=(0.02, 0.25)),
    'perspective': iaa.PerspectiveTransform(scale=(0.05, 0.2)),
}
```

C Datasets

In this work, we consider the following public datasets for handwriting and scene text. Samples from the different datasets are depicted in Fig. 2.

C.1 Handwritten text

For handwriting recognition, we consider three datasets:

- **IAM** [19] handwritten English text dataset, written by 657 different writers. This dataset contains 101,400 correctly segmented words, partitioned into writer independent training, validation and test.
- **CVL** [14] handwritten English text dataset, written by 310 different writers. 27 of the writers wrote 7 texts, and the other 283 writers wrote 5 texts. This dataset contains 84,990 correctly segmented words, partitioned into writer independent training and test. We use the same partitions as in [6, 2], in which the 310 writers are used for training and the additional 27 writers are considered test.
- **RIMES** [8] handwritten French text dataset, written by 1300 different writers. This dataset contains 66,480 correctly segmented words, partitioned into training, validation and test sets that are independent of writers.

Fig. 2: **Dataset samples.** Examples of images from each dataset.

C.2 Scene text

For training the scene text models, we utilized only synthetic datasets:

- **MJSynth** (MJ) [10] a synthetic text in image dataset which contains 9 million word-box images, generated from a lexicon of 90K English words.
- **SynthText** (ST) [9] a synthetic text in image dataset, containing 5.5 million words, designed for scene text detection and recognition.
- **SynthAdd** (SA) [15] only when training SCATTER for scene text, as in the original paper [16], SA was also utilized for training data. This dataset was generated using the same synthetic engine as in ST and contains 1.2 million word box images. SA is used for compensating the lack of non-alphanumeric characters in the training data.

Aligned with many scene text recognition manuscripts (e.g. [24, 3, 15, 16]), we evaluate our models using seven scene text datasets: ICDAR2003, ICDAR2013, IIIT5K, SVT, ICDAR2015, SVTP and CUTE. Those datasets are commonly divided into regular and irregular text according to the text layout.
Regular text datasets are composed of:

- **IIIT5K** [20] consists of 2000 training and 3000 testing images that are cropped from Google image searches.
- **SVT** [25] is a dataset collected from Google Street View images and contains 257 training and 647 testing cropped word-box images.
- **ICDAR2003** [18] contains 867 cropped word-box images for testing.
- **ICDAR2013** [13] contains 848 training and 1,015 testing cropped word-box images.

Irregular text datasets are composed of:

- **ICDAR2015** [12] contains 4,468 training and 2,077 testing cropped word-box images, all captured by Google Glass, without careful positioning or focusing.
- **SVTP** [21] is a dataset collected from Google Street View images and consists of 645 cropped word-box images for testing.
- **CUTE 80** [22] contains 288 cropped word-box images for testing, many of which are curved text images.

D Implementation Details

In our experiments, we utilize four types of architectures. The first is SCATTER, the second and third are two other are variants of the Baek *et al.*[3] framework and the last is AbiNet [5]. All models are trained and tested using the PyTorch framework on a Tesla V100 GPU with 16GB memory.

We follow the training procedure performed in [16]. Accordingly, models are trained using the AdaDelta optimizer with: a decay rate of 0.95, gradient clipping with a magnitude of 5 and batch size of 128. During training, 40% of the input images are augmented by randomly resizing them and adding extra distortion. Models trained on handwriting and scene text datasets are trained for 200k iterations and 600k iterations, respectively. Model selection is performed on the validation set, which for scene text is the union of the IC13, IC15, IIIT5K and SVT training splits and for handwriting is the predefined validation sets. When utilizing SCATTER, all images are resized to 32×128 and are in RGB format. For evaluation, word accuracy measured is case insensitive. We refer the reader for any additional implementation details, both during inference and training, to the original papers [3, 16].

To reproduce the results for AbiNet we downloaded the pretrained models from the official open source implementation (`https://github.com/FangShancheng/ABINet`) and re-ran the fine-tune step. When adding TextAdaIN, we apply it in both phases, in the pre-training step as well as in the fine-tuning step. We use the default configurations as defined in the repository. For handwriting, we use the default configurations but run for 100 epochs and adjust the learning rate schedule to 60, 25, 15.

For TextAdaIN, we use $p = 0.01$ and split images into $K = 5$ windows. In the cases where the number of windows does not divide the width, we use the maximum width that can be divided and ignore the remainder.

E Failure Cases

In Fig. 3, we display failure cases of our method on the IAM dataset. The failure cases are mostly composed of highly cursive text, unclear handwriting styles and ambiguous cases. Adding additional context by employing a line-level approach can assist in rectifying these types of errors, nevertheless, line-level recognition has its own caveats.

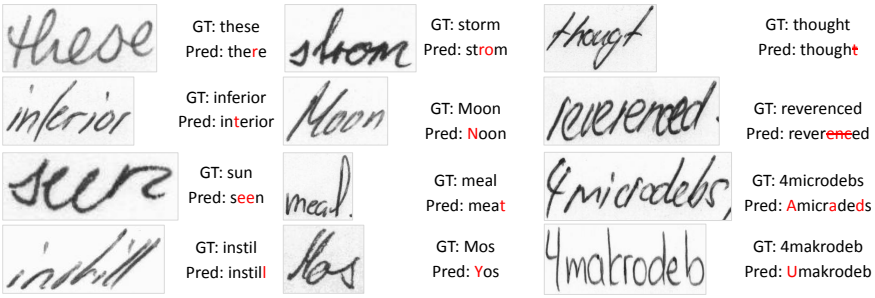

Fig. 3: **Failure cases.** Samples of failure cases on the IAM dataset. GT stands for the ground truth annotation, and Pred is the predicted result. Prediction errors are marked in red, and missing characters are annotated by strike-through.

F The importance of an induced distribution

We show the importance of sampling from an induced distribution, specifically the distribution derived by the representation spaces of natural text images. As shown in Table 1, using Gaussian noise or background images only slightly increases performances. In contrast, TextAdaIN, which samples from the appropriate induced distribution, shows the highest increase in performance.

Table 1: **The importance of the induced distribution.** Best performance is achieved only when injecting distortions from an induced distribution namely, other text images

Injection Method	Donors	Accuracy
Baseline	X	85.7
TextAdaIN	Gaussian Noise	86.1
TextAdaIN	Blank Image	86.2
TextAdaIN	Text Images	**87.3**

G Number of Windows

As mentioned in the main paper, TextAdaIN splits the feature map into windows along the width axis. As the features vary in size at different layers, we define K to represent the number of elements created per sample. Thus, K determines the window size at each layer. Modifying K has several different effects. For example, it controls the granularity level in which the statistics are calculated and modified and the number of donors. Therefore, an optimal value of K can be found to balance the different effects. In Fig. 4,

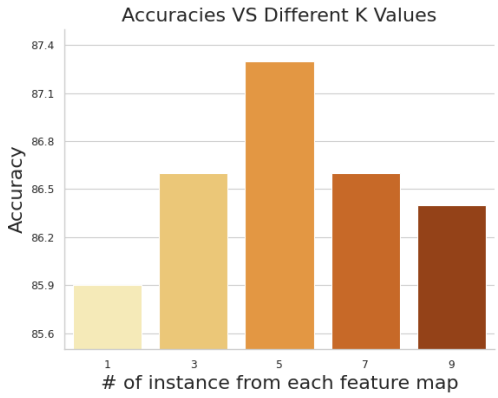

Fig. 4: **Number of windows.** Varying the number of windows extracted per sample has multiple effects, including the granularity level and the number of donor images.

we plot the performance as a function of K. The best result is achieved when using $K = 5$.

H TextAdaIN Pseudo-Code

In this section, we provide pseudo-code for TextAdaIN. The code includes two functions not explicitly implemented: *create_windows_from_tensor*, *revert_windowed_tensor*. The first function represents the mapping:

$$X \in \mathbb{R}^{B \times C \times H \times W} \rightarrow \widehat{X} \in \mathbb{R}^{B \cdot K \times C \times H \times \frac{W}{K}},$$

and the second represents the corresponding inverse mapping. As mentioned in the paper, we do not backpropogate through $\mu_{c,h}(\hat{x}_{\pi(i)}), \sigma_{c,h}(\hat{x}_{\pi(i)})$ and thus, detach is used.

```
def TextAdaIN(x, p=0.01, k=5, eps=1e-4):
    # input x - a pytorch tensor
    if rand() > p:
        return x
    N, C, H, W = x.size()
    # split into windows
    x_hat = create_windows_from_tensor(x,k)
    # calculate statistics
    feat_std = sqrt(x_hat.var(dim=3)+ eps)
    feat_std = feat_std.view(N*k, C, H, 1)
    feat_mean = x_hat.mean(dim=3)
    feat_mean = feat_mean.view(N*k, C, H, 1)
    # perform permutation
    perm_indices = randperm(N*k)
    perm_feat_std = feat_std[perm].detach()
```

```
    perm_feat_mean = feat_mean[perm].detach()
    # normalize
    x_hat = (x - feat_mean) / feat_std
    # swap
    x_hat = x_hat * perm_feat_std
    x_hat += perm_feat_mean
    # merge windows
    x = revert_windowed_tensor(x_hat, k, W)
    return x
```

Bibliography

[1] Aberdam, A., Ganz, R., Mazor, S., Litman, R.: Multimodal semi-supervised learning for text recognition. arXiv preprint arXiv:2205.03873 (2022) 1

[2] Aberdam, A., Litman, R., Tsiper, S., Anschel, O., Slossberg, R., Mazor, S., Manmatha, R., Perona, P.: Sequence-to-sequence contrastive learning for text recognition. arXiv preprint arXiv:2012.10873 (2020) 1, 2

[3] Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What is wrong with scene text recognition model comparisons? dataset and model analysis. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4715–4723 (2019) 1, 3, 4

[4] Cheng, Z., Bai, F., Xu, Y., Zheng, G., Pu, S., Zhou, S.: Focusing attention: Towards accurate text recognition in natural images. In: Proceedings of the IEEE international conference on computer vision. pp. 5076–5084 (2017) 1

[5] *et al.*, F.: Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition 4

[6] Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: Scrabblegan: Semi-supervised varying length handwritten text generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4324–4333 (2020) 2

[7] Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd international conference on Machine learning. pp. 369–376. ACM (2006) 1

[8] Grosicki, E., El Abed, H.: Icdar 2009 handwriting recognition competition. In: 2009 10th International Conference on Document Analysis and Recognition. pp. 1398–1402. IEEE (2009) 2

[9] Gupta, A., Vedaldi, A., Zisserman, A.: Synthetic data for text localisation in natural images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2315–2324 (2016) 3

[10] Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition. arXiv preprint arXiv:1406.2227 (2014) 3

[11] Jung, A.B., Wada, K., Crall, J., Tanaka, S., Graving, J., Reinders, C., Yadav, S., Banerjee, J., Vecsei, G., Kraft, A., Rui, Z., Borovec, J., Vallentin, C., Zhydenko, S., Pfeiffer, K., Cook, B., Fernández, I., De Rainville, F.M., Weng, C.H., Ayala-Acevedo, A., Meudec, R., Laporte, M., et al.: imgaug. https://github.com/aleju/imgaug (2020), online; accessed 01-Feb-2020 1

[12] Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., et al.: Icdar 2015 competition on robust reading. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 1156–1160. IEEE (2015) 4

[13] Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazan, J.A., De Las Heras, L.P.: Icdar 2013 robust reading

competition. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 1484–1493. IEEE (2013) 3
[14] Kleber, F., Fiel, S., Diem, M., Sablatnig, R.: Cvl-database: An off-line database for writer retrieval, writer identification and word spotting. In: 2013 12th international conference on document analysis and recognition. pp. 560–564. IEEE (2013) 2
[15] Li, H., Wang, P., Shen, C., Zhang, G.: Show, attend and read: A simple and strong baseline for irregular text recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 8610–8617 (2019) 3
[16] Litman, R., Anschel, O., Tsiper, S., Litman, R., Mazor, S., Manmatha, R.: Scatter: Selective context attentional scene text recognizer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 1, 3, 4
[17] Liu, W., Chen, C., Wong, K., Su, Z., Han, J.: Star-net: A spatial attention residue network for scene text recognition. In: BMVC (2016) 1
[18] Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: Icdar 2003 robust reading competitions. In: Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. pp. 682–687. Citeseer (2003) 3
[19] Marti, U.V., Bunke, H.: The iam-database: an english sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition **5**(1), 39–46 (2002) 2
[20] Mishra, A., Alahari, K., Jawahar, C.: Scene text recognition using higher order language priors (2012) 3
[21] Quy Phan, T., Shivakumara, P., Tian, S., Lim Tan, C.: Recognizing text with perspective distortion in natural scenes. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 569–576 (2013) 4
[22] Risnumawan, A., Shivakumara, P., Chan, C.S., Tan, C.L.: A robust arbitrary text detection system for natural scene images. Expert Systems with Applications **41**(18), 8027–8048 (2014) 4
[23] Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4168–4176 (2016) 1
[24] Shi, B., Yang, M., Wang, X., Lyu, P., Yao, C., Bai, X.: Aster: An attentional scene text recognizer with flexible rectification. IEEE transactions on pattern analysis and machine intelligence (2018) 3
[25] Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: 2011 International Conference on Computer Vision. pp. 1457–1464. IEEE (2011) 3