Script Identification using Across- and Within-Image Distribution Estimation

Gregory Sell, David Etter, Daniel Garcia-Romero, and Alan McCree
Human Language Technology Center of Excellence
The Johns Hopkins University
Baltimore, USA
{gsell,detter2,dgromero,alan.mccree}@jhu.edu

***Abstract*—In this paper, we apply several modifications to script identification, several of which inspired by techniques from the similar audio task of spoken language recognition. Specifically, we alter the architecture of a convolutional network with global average pooling to include variance pooling as well, we utilize score calibration of the output scores of the network, and we utilize prior distribution estimation to condition the calibrated scores. We show that these methods are effective in script identification, with the use of priors showing especially promising improvements. Furthermore, in the domain of script identification, several additional extensions of distribution estimation are available which consider the distribution within each image, and we demonstrate much larger improvements when employing these extensions. Finally, we also show that an embedding-plus-classifier approach performs similarly to the full network, and so its potential for increased flexibility may be beneficial for future consideration. With all modifications, overall accuracy on the ICDAR 2017 validation dataset increases from 89.7% to 93.6%.**

***Keywords*-script identification, deep learning, prior estimation**

I. Introduction and Related Work

Script identification is an important front-end for optical character recognition (OCR). If an individual recognizer is only trained for particular scripts, it is clearly necessary to first determine which recognizer should be used on the text in a particular image. It is possible that OCR processing will evolve to where a single system can recognize multiple languages and script sets, but in current processing script identification still plays an important role.

Prior to recent years, a variety of approaches were attempted for script identification (e.g., see [1]), but, as has been the case in many fields (including much of computer vision), current methods largely focus on deep learning. As with much of image processing and sequence modeling, one challenge is reducing an input of variable size down to a single common label. Approaches for this include forcing the input image to be a common size so that the network naturally maps down to a label [2], or to use a pooling layer that aggregates across a variable-sized input. In the recent ICDAR 2017 Robust Reading Challenge in Multi-Lingual Scene Text Script Identification (RRC-MLT Task 2) [3], among the official competitors, the top performing system utilized a deep neural network (DNN) with convolutional layers for initial processing and global pooling to aggregate the information [4]. Several variations of that approach were also evaluated, and all performed at similarly high levels, validating the success of the fundamental design. Alternative poolings, such as Spatially Sensitive Pooling [5], have also been suggested with otherwise similar architectures and shown to be successful. Similarly, modified architectures with global average pooling have been explored, such as including recurrent layers prior to pooling [6]. In total, many of the current high-performing sytems utilize a similar process of initial convolutional layers feeding into pooling, and so that is what we will explore in the work that follows.

One advantage of the broad takeover of deep learning across multiple fields is that it is bringing separate application spaces like computer vision and audio processing algorithmically closer together. OCR and automatic speech recognition are both trending towards character-based end-to-end models. Face recognition and speaker recognition are both trending towards similar strategies for DNN embeddings. And so here we propose bringing the fields of script identification and spoken language recognition closer together with an exploration of the benefits of applying techniques drawn from spoken language recognition to script identification. We first consider an architectural modification with variance pooling that is common in state-of-the-art audio embeddings. Second, we apply score calibration learned with heldout data to the output of the network, which converts the output probabilities into true likelihoods. We then utilize distribution estimation to apply priors to the calibrated scores, including new extensions for within-image priors estimated directly from scores or using common script pairings. We also explore the embedding-plus-classifier structure common in spoken language recognition.

II. System Design

We describe below the systems and methods for our experiments. In every trial, a system is presented with a line image x_n (resized to 30 pixels in height while maintaining the aspect ratio and RGB channels) and is asked to predict a label y_n drawn from a script set $\mathcal{S}$, which includes only the seven classes from the RRC-MLT set: Arabic, Bangla, Chinese, Japanese, Korean, Latin, or Symbols. Let us refer to the total number of lines in the dataset as N and the total number of lines in image k as M_k.

A. Network Architecture and Training

The network architecture utilized in the experiments to follow is similar to high-performing systems in the original RRC-MLT Task 2 evaluation, with several two-dimensional convolutional layers followed by a global average pooling, as in [4]. However, unlike the RRC-MLT systems, our convolutional layers increase in dilation as they work up the architecture, similar to a time-delay neural network [7], and the pooling layer includes variance pooling in addition to the average. Both techniques are from recent language recognition work [8], though here implemented in PyTorch and at a reduced size.

Our network begins with a convolutional layer with 128 5x5 filters operating across the three color channels. This is followed by two convolutional layers with 128 3x3 filters and dilation factors of 2 and 3, respectively. After the convolutional layers is an affine layer projecting to size 128 followed by an expansion affine layer to size 512. These 512-dimensional representations are then globally pooled, with either mean pooling or both mean and variance pooling. The pooled statistics are passed through affine layers of size 128, 50, and 128 before the final output size of 7 with a softmax nonlinearity. Rectified linear units and batch normalization [9] are included between all layers.

In our experiments, the network was trained to minimize cross-entropy cost for 200 epochs. For the 10 initial epochs, lines of 30 pixels in height and 200-400 pixels in length are presented to the network in batches of 20. For epochs 11-50, the lines were only 30-200 pixels in length and the batch size was gradually increased to 50 images. Then, for epochs 51-200, half of the line images were also augmented with gaussian blur, speckling, rotations, shifts, or cropping. At all times, Adam optimization [10] is used with a learning rate of 10^{-3} and a weight decay of 10^{-6}.

For each epoch, 10,000 images were randomly selected (with replacement) from the training set for each of the seven scripts, resulting in 70,000 total trials per epoch. This strategy results in a balanced training set for the network, which means that the output posteriors of the system and the likelihoods needed for score calibration (described next) are proportional and can be converted trivially.

B. Score Calibration

Score calibration is the process of converting the output scores of a model into true likelihoods using heldout data matching the expected test domain, and has been a common component in spoken language recognition systems for years [11]. Score calibration is most often necessary for the output of generative models, but here we will explore the value of score calibration on the output posteriors of a DNN, which could aid in generalization or reducing model overfitting, and also could be beneficial in refining scores that were trained with regularization in the cost function.

Score calibration for multi-class language recognition is usually performed as a linear transformation of the log-likelihood $\ell(x_n|y_n) = \log p(x_n|y_n)$. Furthermore, the scaling factor in the linear transform is shared across all classes, and only the bias term is class-specific.

$$\ell'(x_n|y_n) = \alpha\ell(x_n|y_n) + \beta_{y_n} \tag{1}$$

The parameters α and β_{y_n} are learned to minimize multiclass cross entropy for heldout training data, as the errors on the seen training data will be unrealistically low. It is also important that the data used to learn score calibration parameters be a reasonable representation of the test data.

For our experiments, we utilized the publicly-avaliable toolkit `Focal Multi-class` [12].

C. Global Distribution Estimation

It has been shown that additional improvements can be seen in spoken language recognition by estimating the statistics of the classes in the totality of the test data [13]. The use of prior distributions is especially helpful in cases where a system has low confidence. If the output posterior weight is nearly entirely in a particular class, the use of priors is unlikely to overpower that confidence, but in cases where a system is less certain, the use of aggregate prior information can be highly beneficial.

The prior can either be learned from labeled data (with the assumption that the distribution matches the test data), or it can be estimated from unlabeled data. In this work, we will explore both approaches, applying either the train distribution (as seen in Table I) or a distribution estimated from the output scores for the test data.

In the process described in [13], a flat prior is first applied to the calibrated likelihoods, and the resulting posteriors are used to estimate the distribution of each class in the set. The prior estimate is interpolated with a uniform distribution, and so the estimate for iteration i is defined as

$$p_i(y_n) = \alpha p_i'(y_n) + (1-\alpha)p_0(y_n) \tag{2}$$

where p_0 is a uniform distribution and

$$p_i'(y_n) = \frac{1}{N}\sum_n p_{i-1}(y_n|x_n) \tag{3}$$

$$\alpha = \frac{N}{N+R} \tag{4}$$

for some relevance factor R. The posteriors are then updated from the calibrated likelihoods using the new prior.

$$p_i(y_n|x_n) = \frac{p(x_n|y_n)p_i(y_n)}{\sum_{y_n\in\mathcal{S}} p(x_n|y_n)p_i(y_n)} \tag{5}$$

This process is repeated for some fixed number of iterations or until convergence. In our experiments, we always ran 10 iterations (which was found to be consistently sufficient for reasonable convergence) and the relevance factor R was fixed as 1 in all cases.

D. Within-Image Distribution Estimation

A potentially powerful modification of global distribution estimation we propose here is to instead consider the distribution of labels within each image. This is a sensible adjustment, as it stands to reason that the scripts of lines within an image will be drawn from a more specific distribution than the lines from the entire collection, and this more focused domain knowledge should improve performance if it can be effectively estimated.

The process for estimating the within-image distribution is the same as global distribution estimation, except that the processing for image k is performed over only M_k lines instead of N. As a result, a unique prior distribution $p_k(y)$ is estimated for each document. Note that the total line count M_k will now be much smaller than N in the global case, and so the presence of the relevance factor R will have a greater impact on the algorithm. In the case of an image with only one line of text, the prior will be an interpolation halfway between the estimated script likelihoods for that line and the initialization distribution p_0 (uniform or global).

E. Script-Pairing Distribution Estimation

We also propose a further extension of within-image prior probability estimation by incorporating the relationship between script classes, as some scripts are more likely to appear together in documents than others. When estimating the global or within-image prior as described above, this relationship is not considered. So if the network splits the output posterior weight of a particular line image 50-50 between Latin and Arabic, while all other lines in the image are labeled as Chinese with high confidence, we would prefer a prior estimation strategy that considers that Arabic and Chinese are highly unlikely to appear together in the same document, while Latin and Chinese can very plausibly appear together. If we properly account for this relationship, we would determine the confusing line to be much more likely to be Latin than Arabic, while ignoring the relationship would result in equal probabilities of each.

In order to incorporate this information, the prior distribution p_n for line x_n (note that this approach yields a unique prior distribution for each line) is determined by marginalizing the joint distribution of two lines n and m from the same image having any pair of labels. And so, instead of Eq. (3), the prior is estimated from the joint probability averaged over all lines paired with line n.

$$p'_n(y_n) = \frac{1}{M_k - 1} \sum_{m \neq n}^{M_k} \sum_{y_m \in \mathcal{S}} p(y_n, y_m) \tag{6}$$

$$= \frac{1}{M_k - 1} \sum_{m \neq n}^{M_k} \sum_{y_m \in \mathcal{S}} p(y_n | y_m) \hat{p}(y_m) \tag{7}$$

To estimate this distribution, the conditional distribution of a script pairing $p(y_n|y_m)$ can be learned from labeled data or can be estimated from the test scores (only including line pairs from the same source image). For the script probability in the marginalization, we use the output posterior of the network $\hat{p}(y_m) = p(y_m|x_m)$, which can itself incorporate a global or within-image prior distribution.

TABLE I
THE VARIOUS PRIORS LEARNED FROM THE TRAINING DATA. FIRST, THE GLOBAL PRIOR, FOLLOWED BY THE CONDITIONAL PROBABILITIES OF THE SCRIPT IN ONE LINE IN AN IMAGE GIVEN THE SCRIPT OF ANOTHER LINE IN THE SAME IMAGE. THE FINAL ROW (NONE) REFERS TO IMAGES WITH ONLY A SINGLE LINE OF TEXT

		Script Priors (in %)						
		A	B	C	J	K	L	S
	Global	5.4	4.7	3.9	6.7	8.2	69.2	1.8
Given Script	Arabic	61.0	0	0	0	0	34.1	4.9
	Bangla	0	88.3	0	0	0	10.5	1.2
	Chinese	0	0	65.7	0	0	33.9	0.4
	Japanese	0	0	0	65.1	2.0	32.3	0.7
	Korean	0	0	0	2.0	69.2	27.4	1.4
	Latin	0.7	0	0	1.8	1.2	93.9	1.8
	Symbols	5.5	0.1	0	1.3	1.8	81.9	9.5
	None	4.6	11.8	19.7	8.9	7.4	47.5	0

The output of this equation can then be plugged into Eq. (2) as with the other methods. Eq. (5) then estimates a new posterior distribution, which is subsequently used as $\hat{p}(y_m)$ to update the estimate of $p'_n(y_n)$ in Eq. (7).

The conditional distribution $p(y_n|y_m)$ learned from the training data is shown in Table I. The potential for this approach is clear here in several ways. First of all, the conditional distribution when given one of the scripts is quite different than when given a different script, and so there appears to be a great deal of useful information in the pairings. Furthermore, each of the conditional distributions is quite different from the global distribution, also shown in Table I, and so we would similarly expect these distributions to be more appropriate for their particular cases. Finally, it is clear that most of the script pairings are rare, and minimizing the selection of these essentially impossible pairings should be a powerful tool in script identification.

F. Embedding Plus Classifier

It is also common in spoken language recognition to decouple the embedding and classifier. This process was formerly necessary, as the ubiquitous i-vector representation was drawn from a generative model and learned in an unsupervised fashion, and so a separate supervised process was necessary to create class probabilities. In recent years, discriminatively-trained networks have become the standard, but they are still largely used to draw an embedding from an intermediate layer and a classifier is separately learned, though recent work has suggested it may be possible to alter the paradigm [14]. Additional advantages of learning a separate classifier are that new classes can be added without retraining an entire network, and the limited number of parameters could result in easier adaptation in new domains. For completeness, we included this approach in our experiments, using a Gaussian classifier with calibrated scoring [15] on the smaller 50-dimensional penultimate hidden representation in the network.

TABLE II
COMPARING PERFORMANCE OF SYSTEMS WITH AND WITHOUT VARIANCE POOLING. THE COMPARISON OF MULTIPLE NETWORK SIZES SHOWS THAT THE IMPROVEMENT IN PERFORMANCE IS DUE TO THE POOLING MODIFICATION AND NOT TO THE NUMBER OF PARAMETERS.

Pooling	Size	Accuracy (in %)
Mean Only	425k	90.4
Mean+Variance (Smaller)	391k	90.7
Mean+Variance (Larger)	451k	90.8

III. EXPERIMENTAL DESIGN

For all experiments, we utilize the ICDAR 2017 RRC-MLT Task 2 images. However, the bulk of the work here utilizes processing based on grouping the line images from the same source image, and that metadata has not yet been made available for the evaluation set. This information is presumably not yet public in order to maintain the heldout nature of the set for the continuing evaluation of the related text localization Task 1 (and joint labeling in Task 3), but we believe that it is generally not unreasonable to assume the source image for a line of text could be known, and so the methods presented here should be broadly feasible. But, because we require data with the source image known, all our experiments used the provided validation dataset for evaluation, which does connect each line to its source image. We also removed 10% of the training set to use as a heldout set, and the remaining 90% of train was used for training. So, in this design, the validation data was indeed treated as heldout and the experiments are fair and serve our purposes well, but it is important to emphasize this discrepancy so that the numbers presented here are not compared to the performance numbers on the official evaluation dataset.

IV. RESULTS AND DISCUSSION

Accuracies on the validation set (our evaluation set) are shown in Tables II and III. Overall, the full set of alterations suggested here improve performance of the system from 89.7% to 93.6% when the distributions of the train data are known to be applicable to the test data. However, even if this condition is removed and all distributions must be estimated using only the network predictions, the best performance only slightly degrades to 93.3%. Overall, this is a relative improvement of 38% (or 35% in the estimated case), and clearly shows the power of these methods, especially the variations of distribution estimation within images. These results will be examined in detail in the following sections.

A. Variance Pooling

Table II explicitly explores the contributions of variance pooling in addition to mean pooling, which is used in state-of-the-art spoken language recognition while recent script identification methods have only included the latter.

When altering the architecture of the network to accommodate variance pooling, it is impossible to match the parameter allocation exactly. Because two sets of statistics are computed on the pre-pooling layer, we are presented with the option of either increasing the size of the input for the post-pooling affine layer to fit the doubling in size, or to reduce the output size of the pre-pooling affine layer by half (resulting in an equal post-pooling size). The former approach increases the number of network parameters, while the latter decreases it.

In Table II, results for both variations are shown. As can be seen, modest improvements of 0.3-0.4% are achieved with the addition of variance pooling. Furthermore, the improvement is consistent for both the larger and smaller network, suggesting that the variance pooling is in fact providing the gains instead of the increase the parameter size.

In Table III, the mean+variance pooling condition is represented by the smaller network, which was chosen so that the presented improvements are entirely from the algorithm instead of increased parameter counts. A comparison across all conditions shows that the variance pooling consistently improves performance even though the network is smaller in size.

B. Score Calibration

Table III shows accuracies across a number of system designs. The effects of score calibration can be seen by comparing the first and second column for each pooling type. Across all cases, calibration is able to improve performance accuracy modestly but consistently, and, in some cases, the increase is as much as 1-2% (absolute).

However, the main consequence of the better-behaved post-calibration probabilities is that they are better suited for the prediction of and use of more sophisticated prior distributions. If the network is highly confident in the wrong answer, it will hurt distribution estimation, and it will prevent the prior distribution from correcting the mistake. So, a benefit of score calibration is that it both allows for better distributions and more corrections from those distributions. This effect is evident in the performance accuracies in the later rows of Table III, where the calibrated systems consistently outperform the uncalibrated systems often by 1% absolute or more.

C. Global Distribution Estimation

Lines (1) and (2) in Table III show performance when using global prior distributions. In line (1), the distribution is estimated from test scores, while in line (2) the distribution is estimated from the training data (so this is not an oracle distribution, which would use labels from the test data, but does assume train and test are drawn from similar distributions). It is worth noting that the estimated distributions perform only slightly below the distributions learned from the train labels, and also that performance degrades a bit more without score calibration.

D. Within-Image Distribution Estimation

Modifying the distribution estimation to work within each image instead of across images gives a strong improvement of roughly 1% (absolute) in most conditions. This performance can be even further improved by first estimating the global distribution and using it to compute posteriors in the first iteration of the estimation process (line (4)). It is not surprising

TABLE III
ACCURACY (IN %) ON THE ICDAR 2017 RRC-MLT TASK 2 VALIDATION SET (WHICH WE USE AS OUR TEST SET). ITALICIZED LINES APPLY DISTRIBUTIONS LEARNED FROM THE TRAIN SET, SO THESE METHODS ARE ONLY APPLICABLE WHEN THE CLASS DISTRIBUTIONS OF THE TRAIN AND TEST SETS ARE KNOWN TO BE SIMILAR.

				Mean Pooling Only			Mean+Variance Pooling		
Line #	Global Prior	Image Prior	Pairing Prior	Raw Scores	Calibrated Scores	Gaussian Backend	Raw Scores	Calibrated Scores	Gaussian Classifier
(1)	Est	-	-	89.7	90.4	90.5	90.3	90.6	90.6
(2)	*Train*	-	-	*90.0*	*90.4*	*90.5*	*90.5*	*90.7*	*90.7*
(3)	-	Est	-	90.8	91.1	91.4	92.0	92.1	91.2
(4)	Est	Est	-	90.9	91.8	92.2	92.2	92.5	91.9
(5)	*Train*	*Est*	-	*91.0*	*91.9*	*92.2*	*92.2*	*92.5*	*91.9*
(6)	-	-	Est	90.7	91.4	91.8	91.7	92.1	91.5
(7)	-	Est	Est	90.9	92.9	92.9	92.3	93.3	92.9
(8)	Est	Est	Est	91.4	93.0	93.1	92.3	93.3	93.1
(9)	-	-	*Train*	*91.6*	*93.1*	*93.1*	*92.6*	*93.6*	*93.1*

that applying image-specific priors would be effective, but it is noteworthy that the distributions could be estimated from only a handful of scores within a single image. The effectiveness of this approach is likely in part tied to the initial system already performing quite well ($\sim$90% accuracy), and so fewer measurements can still provide a reasonable estimate of the true distribution. This is an interesting shift from global distribution estimation, which is expected to be most helpful for weaker systems, where knowledge of the distribution across a large set can offset an underperforming classifier. The tension between the effectiveness of the within-image priors and the quality of estimate from only a few lines would be an interesting relationship to observe as classifier performance degrades.

E. Script-Pairing Distribution Estimation

Estimating line-specific priors using likely pairings improves performance even further beyond the within-image priors. Estimating the prior with pairings alone perfoms slightly above the performance of within-image priors alone, but slightly lags the performance of estimating within-image priors using the estimated global prior. However, a meaningful boost is seen from using the estimated within-image priors to aid in learning the pairings, increasing performance by over 1% (absolute) in most conditions. A minor additional gain can be seen by first estimating the global prior to use in estimating within-image priors, which are in turn used to estimate the pairings. However, unlike in the previous two cases where the best estimated priors roughly matched the priors learned from the training data, here we see that there are still small gains available by using the relationships learned in the training data. As a result, the best performing system with estimated priors (93.3%) lags the best performing system with learned priors (93.6%), though both represent substantial improvements over the initial system at 89.7%.

F. Gaussian Classifier

Substituting the last several layers of the network with a Gaussian classifier yields performance that is often better than the uncalibrated network, but usually slightly behind the calibrated network. As has been previously discussed, additional benefits of a separate classifier (easier addition of new classes and easier domain adaptation) may make this approach preferable considering there are not significant degradations in performance, but if those particular characteristics are not highly valued, there is much justification here for substituting a complete network for an embedding-plus-classifier design.

G. Error Rate for Number of Lines

Results have clearly demonstrated the value of utilizing versions of within-image prior estimation on the full ICDAR 2017 validation set, and a natural follow-up question is how the effects depend on the amount of total text in the source image. In other words, it would be important to know if the methods are only useful with lots of text. Towards this end, we also analyzed the results based on the number of lines drawn from a source image, shown in Table IV. There are a few noteworthy observations when considering the results in this way.

First of all, somewhat surprisingly, the largest gains are seen in images with a single line, where the distribution in the pairing case is based on other images with only one line. So, the value for those cases is not so much in the within-image prior, but rather in localizing the distribution by which scripts tend to appear alone in images. The reason for this is seen in Table I where the script distribution in images with only one text box is quite different from the global distribution. Even interpolating the global prior with the network likelihoods (as is done in system (5) in Table IV) improves performance, due to the mismatch between the global prior and the prior for single-box images.

We also see the smallest gain in the images with a massive number of lines. This appears to be largely because all systems

TABLE IV
ACCURACY OF THE INCREASINGLY LOCALIZED PRIORS FOR INCREASING NUMBERS OF LINES WITHIN THE SOURCE IMAGE. A GREATER NUMBER OF LINES MEANS A GREATER NUMBER OF MEASUREMENTS USED IN ESTIMATING THE DISTRIBUTIONS.

# of lines	Global (2)	Within (5)	Paired (9)
1	72.5	78.3	82.0
2-5	88.1	90.5	91.8
6-10	91.1	92.0	93.9
11-20	90.6	92.1	93.5
21-50	90.3	92.6	93.4
50-100	91.8	94.0	94.1
100+	96.5	97.3	97.7

perform at a higher level on these lines, presumably indicating that the types of images that would have so many lines have easier text. The relative performance gains are substantial, but due to the high accuracies of the systems, the absolute improvements are fairly modest.

Between the two extremes, the improvements are largely as we would expect. The performance with the global distribution is relatively flat across sizes, while each within-image prior tends to improve with increased numbers of lines.

The script-pairing approach has its strongest advantages over the within-image approach for images with fewer lines. This behavior also makes sense. With a large number of lines, we would expect that the within-image distribution would include all the scripts present in the image, as each script likely appears multiple times. However, when there are only a few lines, then a secondary script may only appear once, meaning that the pairing approach would reasonably be more effective at predicting these scripts from the other more dominant script. It is worth noting that the pairing-based priors are still most effective in all cases, but the advantage diminishes somewhat with increasing number of lines.

V. CONCLUSION

In this work, it was demonstrated that script identification performance can be improved through a number of innovations, most especially by effectively incorporating the aggregate knowledge across all text boxes in a source image. The estimation of distributions within the test images is shown to be quite powerful, and estimating distributions based on common script pairings is even more effective. Additionally, the inclusion of variance pooling improves network performance, even with fewer parameters, and score calibration of the output of the network also yields modest gains. We also showed that altering the system processing to learn a simpler classifier based on DNN embeddings is not necessarily detrimental to performance, and so its greater flexibility may sometimes be preferred. Overall, these methods demonstrated that, even in high-performing systems such as the DNNs trained for script identification, techniques like score calibration and incorporation of prior distributions can provide substantial gains. In this case, the systems were improved from an initial accuracy of 89.7% to a maximum accuracy of 93.6%, an absolute gain of 3.9% and a relative gain of 38%. In the future, these methods could also potentially be applied to other applications in computer vision, such as object detection (where certain objects are likely to appear together) or even face recognition, in cases where multiple faces are present.

REFERENCES

[1] D. Ghosh, T. Dube, and A. Shivaprasad, "Script Recognition - A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2142–2161, December 2010.

[2] J. Zdenek and H. Nakayama, "Bag of Local Convolutional Triplets for Script Identification in Scene Text," in *Proceedings of the International Conference on Document Analysis and Retrieval*, 2017.

[3] N. Nayef, I. Bizid, H. Choi, Y. Feng, D. Karatzas, Z. Luo, U. Pal, C. Rigaud, J. Chazalon, W. Khlif, M. M. Luqman, J.-C. Burie, C. lin Liu, and J.-M. Ogier, "ICDAR2017 Robust Reading Challene on Multi-lingual Scene Text Detection and Script Identification - RRC-MLT," in *Proceedings of the International Conference on Document Analysis and Recognition*, 2017.

[4] Y. Patel, M. Bušta, and J. Matas, "E2E-MLT - an unconstrained end-to-end method for multi-language scene text," 2018, https://arxiv.org/abs/1801.09919.

[5] B. Shi, C. Yao, C. Zhang, X. Guo, F. Huang, and X. Bai, "Automatic script identification in the wild," in *Proceedings of the International Conference on Document Analysis and Retrieval*, 2015.

[6] J. Mei, L. Dai, B. Shi, and X. Bai, "Scene Text Script Identification with Convolutional Recurrent Neural Networks," in *Proceedings of the International Conference on Pattern Recognition*, 2016.

[7] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proceedings of Interspeech*, 2015.

[8] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using x-vectors," in *Proceedings of Odyssey*, 2018.

[9] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Coviariate Shift," in *Proceedings of the International Conference on Machine Learning*, 2015.

[10] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the International Conference for Learning Representations*, 2015.

[11] N. Brümmer and D. A. van Leeuwen, "On calibration of language recognition scores," in *Proceedings of Odyssey*, 2006.

[12] N. Brümmer, "Focal multi-class: Toolkit for evaluation, fusion and calibration of multi-class recognition scores," June 2007.

[13] A. McCree, "Estimating and Exploiting Language Distributions of Unlabeled Data," in *Proceedings of Interspeech*, 2010.

[14] J. Villalba, N. Brümmer, and N. Dehak, "End-to-End versus Embedding Neural Networks for Language Recogntion in Mismatched Conditions," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018.

[15] A. McCree, "Multiclass Discriminative Training of i-vector Language Recognition," in *Proceedings of Odyssey*, 2014.