STEP - Towards Structured Scene-Text Spotting

Sergi Garcia-Bordils[1,2] Dimosthenis Karatzas[1] Marçal Rusiñol[2]
[1]Computer Vision Center, UAB, Spain
[2]AllRead MLT
{sergi.garcia, dimos}@cvc.uab.cat

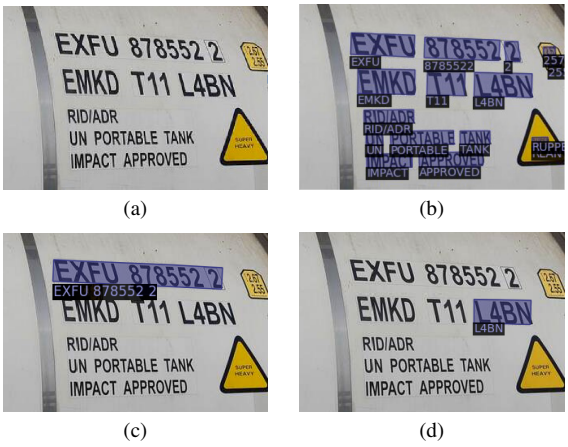
Abstract

We introduce the structured scene-text spotting task, which requires a scene-text OCR system to spot text in the wild according to a query regular expression. Contrary to generic scene text OCR, structured scene-text spotting seeks to dynamically condition both scene text detection and recognition on user-provided regular expressions. To tackle this task, we propose the Structured TExt sPotter (STEP), a model that exploits the provided text structure to guide the OCR process. STEP is able to deal with regular expressions that contain spaces and it is not bound to detection at the word-level granularity. Our approach enables accurate zero-shot structured text spotting in a wide variety of real-world reading scenarios and is solely trained on publicly available data. To demonstrate the effectiveness of our approach, we introduce a new challenging test dataset that contains several types of out-of-vocabulary structured text, reflecting important reading applications of fields such as prices, dates, serial numbers, license plates etc. We demonstrate that STEP can provide specialised OCR performance on demand in all tested scenarios. The code and test dataset are released at https://github.com/CVC-DAG/STEP.

1. Introduction

A lot of textual content that appears in the world around us carries important semantic information, useful for numerous real-world applications. Examples include prices, dates, license plates, serial numbers, consumption readings on utility meters, URLs, telephone numbers, etc. Although scene text detection has advanced significantly over the past decade, current methods still fail to deal satisfactorily with out-of-vocabulary strings, and text in dense configurations, which corresponds exactly to the type of cases of real-life interest.

In this work we propose a new model capable to extract specific text in the wild on demand, as required by the end application. To do so, we exploit the fact that the sought after text has a specific structure to guide both the detection and recognition process dynamically through a query regular expression.



(a) (b) (c) (d)

Figure 1. We propose a model that dynamically conditions both scene text detection and recognition on user provided regular expressions. 1a Image with different types of structured text. 1b Results obtained with TESTR [32]. 1c Result of the proposed method for regular expression "`[A-Za-z]{4}\s\d{6}\s\d`". 1d Results of applying a different regex ("`[A-Za-z]\d[A-Za-z]{2}`").

State of the art scene text recognition methods aim to recognize all text in the scene indiscriminately. Significant progress has been achieved in end-to-end scene text detection and recognition [4, 8, 11, 12, 15, 19, 20, 24, 32] including in challenging scenarios such as curved text [1, 31], text in video [6, 10, 25], or multi-lingual settings [22, 23].

Current methods rely implicitly or explicitly on a language model, which might be acquired by the vocabulary of the training set (implicitly learnt) [5, 30] or explicitly integrated in the recogniser [3]. Numerous recent evidence has demonstrated that such methods tend to over-rely on their language model [5, 30], resulting in higher recognition error rates on out-of-vocabulary text, which is exactly the type of text which is most important for many real-world appli-

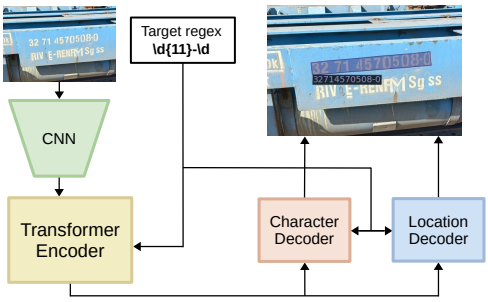


Figure 2. The STEP architecture is composed of a transformer [29] encoder and two character and location decoders, which are guided by the queried regex. The image serves as input to the CNN backbone, while the regex query is used as input by the encoder and decoders.

cations. In addition, all methods perform detection at word-level granularity, while the information of interest might require combinations of various detected tokens. Recovering such information requires heuristic post-processing to combine detected word-level tokens and is prone to prior detection errors. It is possible to train specific systems for extracting specific types of information [7], but it is a cumbersome approach that requires application-specific annotated data, which is not easy to source.

We propose instead to train a single model that can dynamically condition both the detection and recognition stages on a user provided regular expression (regex). The required text structure is provided on demand, and during testing the proposed model is employed in a zero-shot fashion with regular expressions never used during training. The model is capable of spotting only the relevant information in the scene, efficiently suppressing any other distracting text. The regex supported can contain spaces, guiding therefore the detection process from the very beginning to segment the scene text at the required granularity, not imposing any design restrictions to perform detection at the word level. Figure 2 shows an overview of our proposed architecture, the Structured TExt sPotter (STEP).

We compare the proposed system with multiple state of the art methods. We demonstrate that state of the art methods are not capable of correctly detecting and recognising structured text, even if heuristic post-processing is applied on the recognition results. Furthermore, we put forward a new test set reflecting numerous real-life application scenarios where structured text is important and demonstrate the superiority of the proposed method in a zero-shot scenario, spotting regular expressions never seen during training.

The contributions of this paper can be summarized as follows:

- We propose a new structured scene-text spotting task, where methods are expected to detect and recognise text in the wild that respects a dynamically provided query regular expression.
- A challenging test dataset that contains several types of out-of-vocabulary structured text. Each image contains one or more instances of text that respect specific regular expressions. The dataset features also space-separated codes, which poses a particular challenge to generic OCR systems.
- The Structured TExt sPotter (STEP), a network where the detection and the recognition processes are guided with a queried regular expression. The model has been trained on generic, publicly available data, and it is not fine-tuned for any of the test cases.
- We perform comprehensive experimentation and ablation studies, and demonstrate that our approach outperforms state of the art scene-text baselines.

2. Related Work

Our structured scene-text spotting task differs from the classic scene-text detection and recognition paradigm. Generic scene-text datasets feature annotations at arbitrary granularities (the most common being word level), while our task contains text with spaces. Scene-text architectures trained on these datasets can be applied to our task, but require post-processing operations. Since our approach is based on TESTR (a generic scene-text architecture) and trained on public data, in this section we give an overview of the state of the field.

2.1. Scene-Text Detection and Recognition Datasets

Most scene-text datasets feature word-level annotations. They mainly differ on the source of the images, which can be focused (such as in ICDAR13 [10] or incidental text (such as in ICDAR15 [9]). One of the key differences is the type of annotations, which can include rotated quadrilaterals (found for example in ICDAR15 [9], MLT 2017 [23] or MLT2019 [22]), or polygonal annotations (which are used on datasets that focus on irregular text such as Total-Text [1] and CTW1500 [31]). Other datasets like TextOCR [28] and Open Images V5 Text [13] focus on collecting datasets with images that come from large image databases.

All the previously mentioned datasets annotate text at word level. HierText [21], unlike the previously mentioned datasets, contains three hierarchical levels of layout information.The three levels of information are paragraph, line, and word. The line-level information provides adjacency information between the words, which we have used to generate a new dataset featuring spaces. Section 3.2 contains more information about the dataset generation and training procedure.

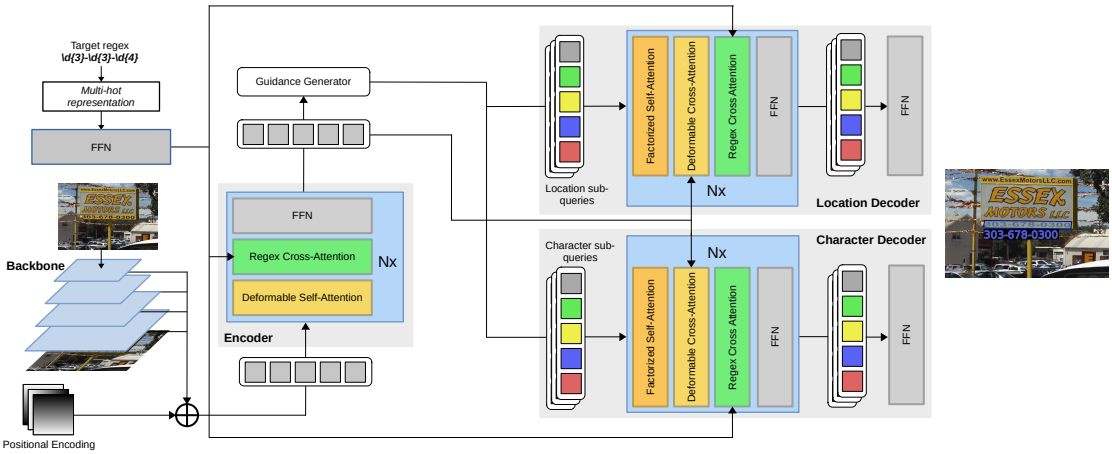


Figure 3. Detailed schematic of STEP, our proposed architecture for structured scene-text detection and recognition based on TESTR [32]. The features extracted by a CNN are the input to a Deformable DETR-like [34] encoder. A cross-attention layer in the encoder combines the image features and the target structure, biasing the guidance generator to generate proposals toward the desired text. Two different branches perform recognition (the character decoding branch) and polygon coordinate regression (the location decoder branch) guided by a cross-attention layer in the character and location decoders.

2.2. Scene-Text Detection and Recognition Architectures

Our proposed task is closely related to scene-text detection and recognition, a field that has attracted active research interest in the last few years. The most common approach has been to use two-stage architectures to perform the detection and recognition. TextBoxes [16] is an example of a two-stage architecture, where an SSD-inspired [17] network performs detections and a CRNN [26] network performs recognition. Since this method was limited to detecting horizontal text, FOTS [18] performs detection using multi-oriented bounding boxes. The authors introduce RoIRotate, a pooling operation that rectifies the visual features horizontally before they get recognized. More recent architectures try to detect arbitrarily shaped scene text. For example, TextDragon [4] predicts text series of quadrangles that follow the text centerline. The ABCNet [19] and ABCNet v2 [20] pipelines use a more unconventional approach by fitting Bezier curves to the text instances.

There has been a recent community trend of utilizing the powerful self-attention mechanism of the transformer [29] architecture. One example of a transformer-based model is TTS [12], which uses a shared transformer encoder-decoder with different decoder heads to perform word recognition, detection, and segmentation. SwinTextSpotter [8] utilizes various transformer-encoder networks to enhance the interaction between detected regions and the recognizer. Some transformer-based models, such as SPTS [24] and DEER [11], can use basic annotations like central keypoints. TESTR [32] also uses an encoder-decoder approach for text detection and recognition. The encoder is based on Deformable DETR [34] detector, while two transformer decoders perform character and polygon decoding. We have based STEP, our approach towards structured text spotting, on this architecture.

2.3. Structured Text Spotting

The domain gap that structured scene-text presents has been an unexplored topic in the community. Approaches such as [7] can learn to recognize structured information (in this case utility meters) but require large amounts of labeled data, which is often limited or outright nonexistent. Related to our idea of exploiting the prior knowledge of the target text, the authors of [27] opt to bias a CNN-LSTM-CTC recognizer network [26] by injecting the regex of the target structure text into the model's decoder. The images are mostly documents and handwritten text. The biasing only takes place on the recognizer, while the localization comes from the line-level information of the dataset. The authors show how this model conditioning reduces spelling mistakes. In our work, we have focused on scene-text spotting, which requires both localization and recognition in natural images. To the best of our knowledge, this is the first attempt at zero-shot, structure-guided scene-text spotting.

3. Methodology

In this section, we describe the architecture and training procedure of Scene TExt sPotter (STEP), our approach towards structured scene-text spotting. Our training and evaluation strategy uses a modified version of HierText to create our training and validation splits.

3.1. Architecture

The STEP architecture is based on TESTR [32], an end-to-end framework for generic text detection and recognition. This architecture is composed of an encoder and two decoder networks based on the transformer [29]. In our architecture, the encoder and decoders have been modified to make them aware of the target structure. This structure is queried as a regular expression, which we represent using a series of multi-hot encoded vectors. Figure 3 shows a detailed overview of STEP.

3.1.1 Text Format Encoding

One of the challenges of this task is to make the network aware of the structure of the target text. STEP's method of representing this structure is based on regular expressions, of which we can represent certain pattern-matching operations. The regex representation is formulated as $\mathbf{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_M)$, where M is the maximum recognition length and each $\mathbf{h}_m$ is a multi-hot encoded vector. Each vector $\mathbf{h}_m$ is defined as $\mathbf{h}_m = (h_{m,1}, \ldots, h_{m,K})$ where $h_{m,k} \in \{0, 1\}$ and K is the number of characters in our character set. An element k represents a particular character of this set. The element $h_{m,k}$ is set to 1 if we know that our target text can have a character with index k in position m. Since this is a multi-hot representation, we can set multiple elements of $\mathbf{h_m}$ to 1.

With this multi-hot encoding we can represent certain regex operands. One of these operations is matching the type of characters at a certain position (in regex, expressions enclosed in brackets "[ ]"). For example, we can encode the pattern "`\b[A-Za-z]{5}`", (any five-letter word). With the multi-hot vectors we can select combinations of different characters, allowing us to encode more general patterns like "`[A-Za-z0-9]{4}''` (any word with a combination of 4 letters or numbers) or more specific ones like "`A\d{2}0`" (any word starting with the letter "A", followed by two numbers, and ending with "0"). Likewise, we can represent the characters that we do not want to match by setting those characters to 0 (in regex, expressed with the bracket expression "`[^ ]`"). For example "`[^1-5]{4}`", any 4-character word that does not contain numbers 1 to 5.

The representation is limited to strings of a fixed number of characters, so the operands "+" or "$\star$" are not supported. We also can not encode the expression "`[0-9]{2-5}`", which represents any number with 2 to 5 digits. As a consequence, we can not query structures of variable length in a single forward pass.

3.1.2 Encoder

The TESTR encoder uses the multi-scale deformable attention module from Deformable DETR [34]. The multi-scale features from the CNN backbone serve as the input to this layer. The deformable attention layer only attends to a small set of keys for each query, reducing the computational complexity of the attention mechanism. The input multi-scale feature maps are defined as $\{\mathbf{x}^l\}_{l=1}^{L}$, where $\mathbf{x}^l \in \mathbb{R}^{C \times H_l \times W_l}$ and L is the level of the feature map. If $\hat{\mathbf{p}}_q \in [0, 1]^2$ are the normalized coordinates of the reference point for a query q, the deformable attention is defined as

$$MSDeformAttn(\mathbf{z}_q, \hat{\mathbf{p}}_q, \{\mathbf{x}^l\}_{l=1}^{L}) = \sum_{m=1}^{M} \mathbf{W}_m \left[\sum_{l=1}^{L} \sum_{k=1}^{K} A_{mlqk} \cdot \mathbf{W}'_m \mathbf{x}^l(\phi_l(\hat{\mathbf{p}}_q) + \Delta \mathbf{p}_{mlqk}) \right] \tag{1}$$

where m, l and k are the attention head, the input feature level and the sampling point, respectively. $\mathbf{A}_{hlk}$ and $\Delta \mathbf{p}_{mlqk}$ are the attention weight and the sampling offset for query element q. $\phi_l(\hat{\mathbf{p}}_q)$ performs a mapping from normalized image coordinates to its location in the l-th level of the feature map. $\mathbf{W}_m$ and $\mathbf{W}'_m$ are trainable matrices.

Like in TESTR, STEP uses the Two-Stage version of the Deformable DETR. In the Two-Scale variant, the guidance generator at the output of the encoder generates coarse bounding box proposals as the first stage. These generated proposals serve as the initialization of the object queries in the decoders, the second phase. Each pixel of the multi-scale feature map is used to generate a proposal, but only the top-scoring bounding boxes are picked. The generation of the bounding box proposals is supervised with an intermediate classification loss and an IoU loss.

In the regular TESTR, the bounding boxes should be generated over the areas of the image that contain text. Since the model is often trained at word granularity, it is easy for the encoder to come up with reasonably good proposals. In our problem, however, we can not directly use this approach, since the generated boxes should be over areas of text that follow the target structure. This requires the encoder to somehow be aware of the regex before generating the proposals. In STEP, we have added an additional multi-head cross-attention layer in each of the encoder layers. This cross-attention uses the multi-scale image features as the queries, and the encoded regex $\mathbf{H}$ as the key and values. This layer conditions the encoder to generate proposals over areas of text that follow the queried regex.

3.1.3 Decoders

STEP follows the idea of TESTR of tackling text detection and recognition as learning to predict tuples of points and characters. If K is the number of proposals of the guidance generator and i is the index of each proposal, the model learns to predict the tuple $Y = \{(\mathbf{P}^{(i)}, \mathbf{C}^{(i)}\}_{i=1}^{K}$, where

$\mathbf{P}^{(i)} = (p_1^{(i)}, \ldots, p_N^{(i)})$ are the N coordinates of the predicted polygon and $\mathbf{C}^{(i)} = (c_1^{(i)}, \ldots, c_M^{(i)})$ are the M characters of the predicted text. The two sets of elements of the tuple are predicted by the location and character decoders.

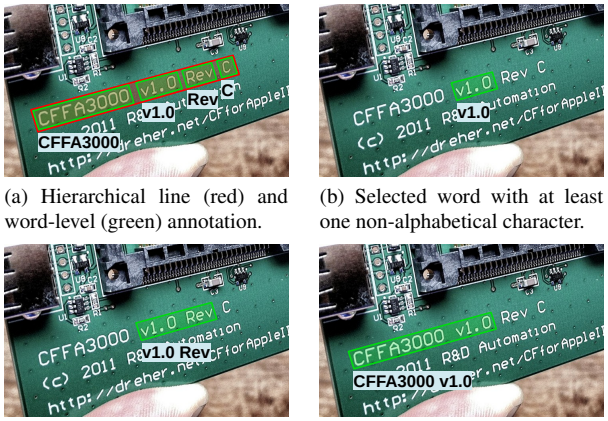
The character decoder extends each query to M sub-queries, each sub-query is a character of the recognition $\mathbf{C}^{(i)} = (c_1^{(i)}, \ldots, c_M^{(i)})$. The decoder is composed of a deformable cross-attention layer with the image features and factorized self-attention layers. The factorized self-attention, inspired by [2], includes an intra-attention layer between the elements of $\mathbf{C}^{(i)}$ and an inter-attention between the characters c_j across different words. A classification layer predicts the final class of each sub-query. Additionally, STEP adds a cross-attention layer between the elements of $\mathbf{C}^{(i)}$ and the embedded regex expression $\mathbf{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_M)$. The character sub-queries serve as the queries while vectors of the regex serve as the keys and values. This cross-attention guides the recognition process and reduces spelling mistakes. Similarly, the location decoder extends each instance query with N sub-queries, and each sub-query is a control point of the polygon $\mathbf{P}^{(i)} = (p_1^{(i)}, \ldots, p_N^{(i)})$. Like in the character decoder, it is composed of a deformable cross-attention layer with the image features and factorized self-attention layers. We also add a cross-attention layer in between the sub-query points $\mathbf{P}^{(i)}$ of each polygon and the vectors of the regex $\mathbf{H}$.

STEP also differs in the approach to calculating each proposal's objectness score. Vanilla TESTR uses each location sub-query to predict a confidence score, where the average is the final score of the instance. In regular scene-text detection, visual appearance may suffice to determine if a proposal overlaps with a text instance. However, in structured text spotting, the contents of the detected region are also important to ascertain the validity of a proposal. The text within the region must adhere to the queried structure, which means that transcription information should also be factored in. To address this, STEP utilizes both location and character sub-queries to produce the confidence score.

3.2. Model Training

Classic scene text datasets feature word-level annotations. Our objective is to build an OCR system that is capable of detecting and recognizing text with arbitrary structures, which can include spaces. The HierText [21] dataset is a particular case among scene-text datasets. This dataset features three levels of hierarchical annotations; paragraph, line, and word level. We have modified this dataset to create varied and challenging training and evaluation splits, which includes spaces.

To reduce the bias towards in-vocabulary words (as defined by [30], words seen during the training phase) in our modified HierText dataset, we only kept the instances that contain at least one non-alphabetical character. We also use the line-level information to merge adjacent neighbors and create new annotations with spaces. The process includes merging the polygons and both captions, which are concatenated with a space in-between. The merging strategy is shown in Figure 4.



(a) Hierarchical line (red) and word-level (green) annotation.

(b) Selected word with at least one non-alphabetical character.

(c) Merging of the selected word and its left neighbor.

(d) Merging of the selected word and its right neighbor.

Figure 4. Our HierText-derived dataset uses line and word-level annotations to create new annotations with spaces. Starting off from a single annotated line (Figure 4a), we keep all the words that contain at least one non-alphabetical character (Figure 4b). Additionally, we also try to create new annotations by merging the selected annotation with its adjacent words. In Figures 4c and 4d, we have merged the polygons of the word "v1.0" with its two adjacent words. The final captions are the two sub-captions separated by a space.

The training pipeline of the network follows a similar strategy as other scene-text models, with the difference that our ground truth only contains the instances that match the query. Each time the dataloader samples an image, we select one of the ground truth instances at random (the image might contain numerous), which we use to generate the regex representation $\mathbf{H}$. For each character m of the selected word, we generate its vector $\mathbf{h}_m$. The elements of $\mathbf{h}_m$ are set to 1 depending on the type of the character (letter, number, space, etc.). We can also randomly set just one element of $\mathbf{h}_m$ to 1 in order to force the specific character m in that position. Since multiple text instances can match the generated query, the rest of the words of the image are compared against the generated regex. The matching instances are included in the ground truth. The disadvantage of this approach is that we need more training iterations in order to see all the ground truth instances of the training set.

3.2.1 Training Details

We use the TESTR pre-trained weights provided by the authors to initialize STEP, all the layers that are not present on the original TESTR are randomly initialized. These pre-

trained weights used a mixture of SynthText 150k (coming from [19], MLT 17 [23] and Total-Text [1]. The initial learning rate of the network is 10^{-4}, and is decayed by a factor of 0.1 at 60k and 200k steps, the model is trained for a total of 300k steps. Like in TESTR, the learning rates of the backbone and linear projections of the reference points and sampling offsets are scaled by a factor of 0.1. The optimizer is AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay of 10^{-4}. The batch size used is 6 images and the training takes around 2 days in two RTX 6000 Ada Generation GPUs.

3.3. Model Evaluation

We use the modified validation split of HierText to evaluate both our model and the generic baselines. Like in the training split, the evaluation contains spaced text. The evaluation protocol provides the regex of the target text to each method, which has to locate and transcribe all the matching instances. This format contains the type of each character (letter, number, space, etc.) and its length. For example, for the string target "Abcd 123-1", the provided regex is "`[a-zA-Z]{4}\s\d{3}-\d`". Methods can make use of this information to guide the network or perform post-processing operations. We used the classical precision and recall metrics used in object detection to evaluate on this split. Instances must have an intersection over union of over 0.5 with a ground truth instance to be considered a localization match. When the model is being evaluated End-To-End, the transcription of the proposal and the ground truth instance must be the same.

4. Structured Scene-Text Dataset

To better display the zero-shot capabilities of our approach, we introduce a new scene-text test dataset that puts the focus on out-of-vocabulary structured scene-text. Our dataset includes 836 images where 6 types of formatted text have been annotated. Much of the text contains multiple spaces and does not follow text found in any vocabulary (as opposed to generic scene-text datasets). Since the format of each code is known, methods can make use of this prior information to condition the network or use it in post-processing operations (just like in our evaluation protocol). Figure 5 shows qualitative examples of some of the formats included in this split. Section F of the supplementary material contains further information about the images and text featured in the test split.

With the exception of license plates and phone numbers, all images have been collected by us. The images with license plates come from the UFPR-ALPR dataset [14]. This dataset contains 150 sequences of 30 images, each one of the sequences features a vehicle with a license plate. The 150 sequences are divided into test, train, and validation. In our test dataset we have kept the first image of every sequence of the three splits, and we have discarded all the vertical license plates, ending up with a total of 121 images. The images with phone numbers come from the Ubertext [33] dataset, a large-scale scene-text dataset sourced from the Bing Maps Streetside program where each text instance is labeled with a category (such as "license plate" or "street number"). We have collected a total of 109 images that contain at least one phone number.

5. Experiments

We compare our approach with TESTR [32], SwinTextSpotter [8] and ABCNet v2 [20], three generic state-of-art scene-text models we use as baselines. All these models have been fine-tuned on the vanilla HierText dataset until convergence. In section C of the supplementary material we provide the training details for each one of the baselines. In order to deal with spaced text (which is not present in vanilla HierText), we applied post-processing operations on the detected areas of the image.

5.1. HierText Evaluation Dataset

Following the protocol described in section 3.3, we evaluated the baselines and our structure-guided architecture. We use the provided regex to guide the detection and recognition of our model. On the generic baselines, we use this information to perform post-processing operations on the detected text. These post-processing operations include merging instances and filtering non-matching text. This instance merging tries to join detections in case the queried regex features a space. Section A of the supplementary provides more details about the merging procedure.

Table 1 shows the End-To-End and Detection validation results for both the baselines and our method. On End-To-End, our model displays higher precision than the baselines and much higher recall (12% more than the TESTR baseline), which results in a higher F-score. Biasing the model with the structure of the target also reduces the number of spelling mistakes, as shown in the average edit distance. Our model also obtains better detection results than the baselines. One major disadvantage of the generic methods is that since we are using the detection's recognition to filter out irrelevant text, some spelling mistakes can harm the detection performance. Mistaking a number for a letter can make the filtering process discard a valid detection, given that it does not follow the provided structure.

5.2. Structured Scene-Text Dataset

This section presents the results on the structured text dataset introduced in section 4. Following the same approach as in the evaluation set, we provide the format of the target text for the different methods tested. As in the evaluation split, the detections of the baselines are merged if the queried regex contains one or more spaces.

Model	End-To-End				Detection		
	Precision	Recall	F-score	Avg. ED	Precision	Recall	F-score
ABCnet v2 [20]	0.72	0.31	0.43	0.26	**0.9**	0.27	0.42
SwinTS [8]	0.67	0.27	0.39	0.22	0.80	0.32	0.46
TESTR [32]	0.72	0.50	0.59	0.19	0.87	0.51	0.64
STEP	**0.78**	**0.64**	**0.71**	**0.13**	0.86	**0.69**	**0.76**

Table 1. End-To-End and Detection results on our HierText-based evaluation dataset.

Model	Post-Processing	BIC	UIC	TARE	Phone Num.	Tonnage	License Plate	Avg. ED
ABCnet v2 [20]	✗	0.01	0.03	0.41	0.25	0.34	0.22	2.03
ABCnet v2 [20]	✓	0.15	0.12	0.47	0.32	0.33	0.33	1.87
SwintTS [8]	✗	0.0	0.02	0.49	0.4	0.37	0.24	1.63
SwintTS [8]	✓	0.36	0.12	0.59	0.45	0.38	0.38	1.30
TESTR [32]	✗	0.03	0.07	0.43	0.58	0.40	0.18	0.46
TESTR [32]	✓	0.43	**0.26**	0.62	0.65	0.39	0.29	0.70
STEP	-	**0.75**	0.24	**0.86**	**0.68**	**0.72**	**0.55**	**0.25**

Table 2. End-to-end results on the test split. Each cell of the table shows the final F-score of each method for a particular code. The last column shows the average edit distance over all the instances.

Tables 2 and 3 show the End-To-End and Detection results on the test dataset. Each cell showcases the F-score for each method and code type. We include the baselines with and without post-processing operations. When the baselines do not use post-processing operations, the results are considerably worse on formats with spaces such as the BIC and UIC codes. Our method obtains considerably better scores in both tasks for all the code types except in end-to-end UIC codes, probably due to the difficulty to recognize long sequences (although still obtains higher results in the detection task).

5.3. Qualitative Examples

Figure 5 shows qualitative examples of TESTR and STEP on different structured codes of our test dataset. The TESTR results are shown without any post-processing operations except for removing irrelevant text. The BIC and UIC codes get fragmented into different detections by TESTR, something that is not an issue with STEP. Numerous text fragmentations increase the probability of missing a part of the code, so removing this problem helps increase the recall of our method. Using the format of the code also allows our network to make fewer spelling mistakes, especially those that are related to the structure of the text. In the TARE and phone number examples, TESTR has read the wrong number of digits. Our model has less chance of getting the structure of the text wrong thanks to the cross-attention layers in the character decoder. We also avoid confusing letters with numbers as shown in the license plate example, where TESTR has mixed the letter "I" for the number "1".

5.4. Ablation Studies

In this section we ablate the main architectural changes performed to TESTR. The studies have been conducted on the Hier-Text-based validation dataset.

5.4.1 Model Confidence

In section 3.1.3 we describe how STEP uses the location and character sub-queries of an instance to calculate its classification score. In Table 4 we compare the impact on the model performance of using the character and location sub-queries. When the model uses only the character sub-queries, the F-score improves 6% points with respect to the location-only baseline. When we use both the score is slightly improved by a 1%. The final version of STEP uses both modalities to calculate the detection confidence.

5.4.2 Regex Cross-Attention

STEP features three cross-attention layers with the regex, one on the encoder and one in each of the two decoders. The purpose of these layers is to bias the guidance generator, the location decoder, and the character decoder. In this ablation experiment, we show the impact of removing or adding these layers to the encoder and two decoders of the network. Results are presented in Table 5. As seen in the results, the encoder cross-attention is necessary to generate quality proposals. Without this layer, the guidance generator and the rest of the network are completely unaware of the structure. Adding the cross-attention in the character decoder reduces spelling mistakes and increases the F-score

Model	Post-Processing	BIC	UIC	TARE	Phone Num.	Tonnage	License Plate
ABCnet v2 [20]	✗	0.03	0.08	0.46	0.54	0.35	0.37
ABCnet v2 [20]	✓	0.23	0.19	0.61	0.60	0.35	0.56
SwintTS [8]	✗	0.01	0.10	0.56	0.65	0.37	0.46
SwintTS [8]	✓	0.42	0.2	0.67	0.73	0.38	0.65
TESTR [32]	✗	0.04	0.08	0.43	0.67	0.41	0.21
TESTR [32]	✓	0.47	0.27	0.64	0.72	0.40	0.37
STEP	-	**0.9**	**0.71**	**0.94**	**0.83**	**0.74**	**0.79**

Table 3. Detection results on the test split. Each cell of the table shows the final F-score of each method for a particular code.

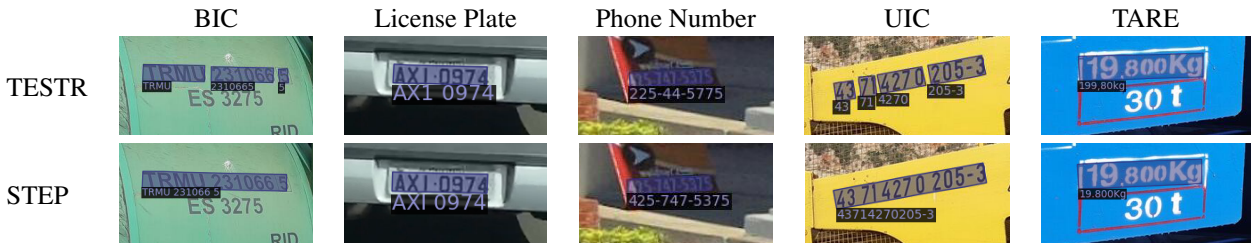


Figure 5. Qualitative results of TESTR and STEP on some examples from the test dataset.

Character	Location	P	R	Fs	ED
✗	✓	0.73	0.52	0.61	**0.09**
✓	✗	**0.78**	0.60	0.67	0.11
✓	✓	**0.78**	**0.64**	**0.71**	0.13

Table 4. Results of using the character and location sub-queries on the HierText-based evaluation dataset.

Enc.	Char.	Loc.	P	R	Fs	ED
✗	✗	✗	**0.87**	0.12	0.21	**0.05**
✓	✗	✗	0.76	0.56	0.64	0.08
✓	✓	✗	0.77	0.61	0.68	0.15
✓	✓	✓	0.78	**0.64**	**0.71**	0.13

Table 5. Impact of including the regex cross-attention layers on the encoder and decoders of the network. The results reported are End-To-End on the HierText-based validation set.

by 4%. Finally, adding the cross-attention layer in the location decoder helps boost the F-score by 3 additional points.

6. Discussion

6.1. Limitations

We have shown that our approach to structured scene-text can effectively locate text with known formats in a zero-shot manner. However, our model has some limitations, and could still be further improved beyond quantitative metrics. In the first place, our architecture is not capable of dealing with more than one structure query at a time. Multiple queries require to do multiple forward passes. This is not a limitation of generic OCR systems, since they always produce the same readings for a given image. Our representation of the regex is also limited to strings of a fixed length, so we can not use regex operators such as "+" or "⋆".

6.2. Conclusions

In this paper, we have proposed the task of structured scene-text spotting, a novel structured-text test dataset, and STEP, our approach to tackle this problem. We have shown how by providing the structure to our model we can successfully guide the text-spotting process. The proposed method effectively removes detection fragmentations and reduces spelling mistakes. With the training strategy proposed, the network can entirely be trained on public data and generalize well on unseen data. This approach has been shown to be superior to using generic scene-text detection and recognition systems coupled with post-processing operations.

Acknowledgements This work has been supported by grants PDC2021-121512-I00, PID2020-116298GB-I00 and PLEC2021-007850 funded by the European Union NextGenerationEU/PRTR and MCIN/AEI/10.13039/501100011033; the EU Lighthouse on Safe and Secure AI - ELSA funded by European Union's Horizon Europe programme under grant agreement No 101070617; the Spanish Projects NEOTEC SNEO-20211172 from CDTI and CREATEC-CV IMCBTA/2020/46; grant Torres Quevedo PTQ2019-010662; and the Industrial Doctorate programme of the Catalan Government (2020 DI 058).

References

[1] Chee Kheng Ch'ng and Chee Seng Chan. Total-text: A comprehensive dataset for scene text detection and recognition. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 935–942. IEEE, 2017. 1, 2, 6

[2] Qi Dong, Zhuowen Tu, Haofu Liao, Yuting Zhang, Vijay Mahadevan, and Stefano Soatto. Visual relationship detection using part-and-sum transformers with composite queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3550–3559, 2021. 5

[3] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7098–7107, 2021. 1

[4] Wei Feng, Wenhao He, Fei Yin, Xu-Yao Zhang, and Cheng-Lin Liu. Textdragon: An end-to-end framework for arbitrary shaped text spotting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9076–9085, 2019. 1, 3

[5] Sergi Garcia-Bordils, Andrés Mafla, Ali Furkan Biten, Oren Nuriel, Aviad Aberdam, Shai Mazor, Ron Litman, and Dimosthenis Karatzas. Out-of-vocabulary challenge report. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 359–375. Springer, 2023. 1

[6] Sergi Garcia-Bordils, George Tom, Sangeeth Reddy, Minesh Mathew, Marçal Rusiñol, CV Jawahar, and Dimosthenis Karatzas. Read while you drive-multilingual text tracking on the road. In *Document Analysis Systems: 15th IAPR International Workshop, DAS 2022, La Rochelle, France, May 22–25, 2022, Proceedings*, pages 756–770. Springer, 2022. 1

[7] Luís Gómez, Marçal Rusinol, and Dimosthenis Karatzas. Cutting sayre's knot: reading scene text without segmentation. application to utility meters. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 97–102. IEEE, 2018. 2, 3

[8] Mingxin Huang, Yuliang Liu, Zhenghao Peng, Chongyu Liu, Dahua Lin, Shenggao Zhu, Nicholas Yuan, Kai Ding, and Lianwen Jin. Swintextspotter: Scene text spotting via better synergy between text detection and text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4593–4603, 2022. 1, 3, 6, 7, 8

[9] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *2015 13th international conference on document analysis and recognition (ICDAR)*, pages 1156–1160. IEEE, 2015. 2

[10] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *2013 12th international conference on document analysis and recognition*, pages 1484–1493. IEEE, 2013. 1, 2

[11] Seonghyeon Kim, Seung Shin, Yoonsik Kim, Han-Cheol Cho, Taeho Kil, Jaeheung Surh, Seunghyun Park, Bado Lee, and Youngmin Baek. Deer: Detection-agnostic end-to-end recognizer for scene text spotting. *arXiv preprint arXiv:2203.05122*, 2022. 1, 3

[12] Yair Kittenplon, Inbal Lavi, Sharon Fogel, Yarin Bar, R Manmatha, and Pietro Perona. Towards weakly-supervised text spotting using a multi-task transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4613, 2022. 1, 3

[13] Ilya Krylov, Sergei Nosov, and Vladislav Sovrasov. Open images v5 text annotation and yet another mask text spotter. In *Asian Conference on Machine Learning*, pages 379–389. PMLR, 2021. 2

[14] Rayson Laroca, Evair Severo, Luiz A Zanlorensi, Luiz S Oliveira, Gabriel Resende Gonçalves, William Robson Schwartz, and David Menotti. A robust real-time automatic license plate recognition based on the yolo detector. In *2018 international joint conference on neural networks (ijcnn)*, pages 1–10. IEEE, 2018. 6

[15] Minghui Liao, Guan Pang, Jing Huang, Tal Hassner, and Xiang Bai. Mask textspotter v3: Segmentation proposal network for robust scene text spotting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 706–722. Springer, 2020. 1

[16] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017. 3

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016. 3

[18] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. Fots: Fast oriented text spotting with a unified network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5676–5685, 2018. 3

[19] Yuliang Liu, Hao Chen, Chunhua Shen, Tong He, Lianwen Jin, and Liangwei Wang. Abcnet: Real-time scene text spotting with adaptive bezier-curve network. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9809–9818, 2020. 1, 3, 6

[20] Yuliang Liu, Chunhua Shen, Lianwen Jin, Tong He, Peng Chen, Chongyu Liu, and Hao Chen. Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8048–8064, 2021. 1, 3, 6, 7, 8

[21] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards

end-to-end unified scene text detection and layout analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1049–1059, 2022. 2, 5

[22] Nibal Nayef, Yash Patel, Michal Busta, Pinaki Nath Chowdhury, Dimosthenis Karatzas, Wafa Khlif, Jiri Matas, Umapada Pal, Jean-Christophe Burie, Cheng-lin Liu, et al. Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition—rrc-mlt-2019. In *2019 International conference on document analysis and recognition (ICDAR)*, pages 1582–1587. IEEE, 2019. 1, 2

[23] Nibal Nayef, Fei Yin, Imen Bizid, Hyunsoo Choi, Yuan Feng, Dimosthenis Karatzas, Zhenbo Luo, Umapada Pal, Christophe Rigaud, Joseph Chazalon, et al. Icdar2017 robust reading challenge on multi-lingual scene text detection and script identification-rrc-mlt. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1454–1459. IEEE, 2017. 1, 2, 6

[24] Dezhi Peng, Xinyu Wang, Yuliang Liu, Jiaxin Zhang, Mingxin Huang, Songxuan Lai, Jing Li, Shenggao Zhu, Dahua Lin, Chunhua Shen, et al. Spts: single-point text spotting. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4272–4281, 2022. 1, 3

[25] Sangeeth Reddy, Minesh Mathew, Lluis Gomez, Marçal Rusinol, Dimosthenis Karatzas, and CV Jawahar. Roadtext-1k: Text detection & recognition dataset for driving videos. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11074–11080. IEEE, 2020. 1

[26] 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, 2016. 3

[27] Baoguang Shi, Wenfeng Cheng, Yijuan Lu, Cha Zhang, and Dinei Florencio. Improving structured text recognition with regular expression biasing. *arXiv preprint arXiv:2111.06738*, 2021. 3

[28] Amanpreet Singh, Guan Pang, Mandy Toh, Jing Huang, Wojciech Galuba, and Tal Hassner. Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8802–8812, 2021. 2

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 3, 4

[30] Zhaoyi Wan, Jielei Zhang, Liang Zhang, Jiebo Luo, and Cong Yao. On vocabulary reliance in scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11425–11434, 2020. 1, 5

[31] Liu Yuliang, Jin Lianwen, Zhang Shuaitao, and Zhang Sheng. Detecting curve text in the wild: New dataset and new solution. *arXiv preprint arXiv:1712.02170*, 2017. 1, 2

[32] Xiang Zhang, Yongwen Su, Subarna Tripathi, and Zhuowen Tu. Text spotting transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9519–9528, 2022. 1, 3, 4, 6, 7, 8

[33] Ying Zhang, Lionel Gueguen, Ilya Zharkov, Peter Zhang, Keith Seifert, and Ben Kadlec. Uber-text: A large-scale dataset for optical character recognition from street-level imagery. In *SUNw: Scene Understanding Workshop-CVPR*, volume 2017, page 5, 2017. 6

[34] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3, 4