Visual Re-ranking with Natural Language Understanding for Text Spotting

Ahmed Sabir[1], Francesc Moreno-Noguer[2] and Lluís Padró[1]

[1] TALP Research Center, Universitat Politècnica de Catalunya, Barcelona, Spain
[2] Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Barcelona, Spain
asabir@cs.upc.edu, fmoreno@iri.upc.edu, padro@cs.upc.edu

Abstract. Many scene text recognition approaches are based on purely visual information and ignore the semantic relation between scene and text. In this paper, we tackle this problem from natural language processing perspective to fill the gap between language and vision. We propose a post-processing approach to improve scene text recognition accuracy by using occurrence probabilities of words (unigram language model), and the semantic correlation between scene and text. For this, we initially rely on an off-the-shelf deep neural network, already trained with large amount of data, which provides a series of text hypotheses per input image. These hypotheses are then re-ranked using word frequencies and semantic relatedness with objects or scenes in the image. As a result of this combination, the performance of the original network is boosted with almost no additional cost. We validate our approach on ICDAR'17 dataset.

1 Introduction

Machine reading has shown a remarkable progress in Optical Character Recognition systems (OCR). However, the success of most OCR systems is restricted to simple-background and properly aligned documents, while text in many real images is affected by a number of artifacts including partial occlusion, distorted perspective and complex backgrounds. In short, developing OCR systems able to read text in the wild is still an open problem. In the computer vision community, this problem is known as *Text Spotting*. However, while state-of-the-art computer vision algorithms have shown remarkable results in recognizing object instances in these images, understanding and recognizing the included text in a robust manner is far from being considered a solved problem.

Text spotting pipelines address the end-to-end problem of detecting and recognizing text in unrestricted images (traffic signs, advertisements, brands in clothing, etc.). The problem is usually split in two phases: 1) *text detection stage*, to estimate the bounding box around the candidate word in the image and 2) *text recognition stage*, to identify the text inside the bounding boxes. In this work we focus on the second stage, an introduce a simple but efficient post-processing approach based on Natural Language Processing (NLP) techniques.

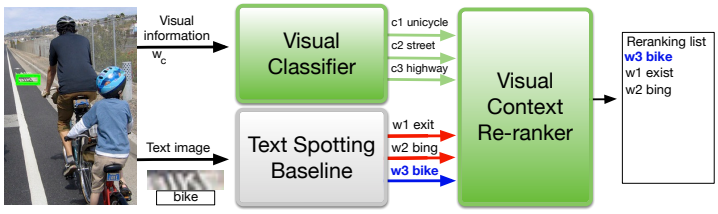

Fig. 1. Overview of the system pipeline. A re-ranking post-process using visual context information to re-rank the potential candidate word based on the semantic relatedness with the context in the image (where the text is located). In the example of the figure, the word *bike* has been re-ranked thanks to the detected visuals (w_c) *unicycle, street, highway.*

There exist two main approaches to perform text recognition in the wild. First, lexicon-based methods, where the system learns to recognize words in a pre-defined dictionary. Second, lexicon-free, unconstrained recognition methods, that aim at predicting character sequences.

In this paper, we propose an approach that intends to fill the gap between language and vision for the scene text recognition problem. Most recent state-of-the-art works focus on automatically detecting and recognizing text in unrestricted images from a purely computer vision perspective. In this work, we tackle the same problem but also leveraging on NLP techniques. Our approach seeks to integrate prior information to the text spotting pipeline. This prior information biases the initial ranking of candidate words suggested by a deep neural network, either lexicon based or not. The final re-ranking is based on the word frequency and on the semantic relatedness between the candidate words and the information in the image.

Figure 1 shows an example where the candidate word *bike* is re-ranked thanks to the visual context information *unicycle, street, highway* detected by a visual classifier. This is a clear example that illustrates the main idea of our approach.

Our main contributions include several post-processing methods based on NLP techniques such as word frequencies and semantic relatedness which are typically exploited in NLP problems but less common in computer vision ones. We show that by introducing a candidate re-ranker based on word frequencies and semantic distance between candidate words and objects in the image, the performance of an off-the-shelf deep neural network can be improved without the need to perform additional training or tuning. In addition, thanks to the inclusion of the unigram probabilities, we overcome the baseline limitation of false detection of short words of [1,2].

The rest of the paper is organized as follows: Sections 2 and 3 describe related work and our proposed pipeline. Sections 4 and 5 introduce the external prior knowledge we use, and how it is combined. Section 6 presents experimental validation of our approach on a publicly available standard dataset. Finally, Sections 7 and 8 summarize the result and specifies future work.

2 Related Work

Text spotting (or end-to-end text recognition), refers to the problem of automatically detecting and recognizing text in images in the wild. Text spotting may be tackled by either a lexicon-based or a lexicon-free perspective. Lexicon-based recognition methods use a pre-defined dictionary as a reference to guide the recognition. Lexicon-free methods (or unconstrained recognition techniques), predict character sequences without relying on any dictionary. The first lexicon-free text spotting system was proposed by [3]. The system extracted character candidates via maximally stable extremal regions (MSER) and eliminated non-textual ones through a trained classifier. The remaining candidates were fed into a character recognition module, trained using a large amount of synthetic data. More recently, several deep learning alternatives have been proposed. For instance, PhotoOCR [4] uses a Deep Neural Network (DNN) that performs end-to-end text spotting using histograms of oriented gradients as input of the network. It is a lexicon-free system able to read characters in uncontrolled conditions. The final word re-ranking is performed by means of two language models, namely a character and an N-gram language model. This approach combined two language models, a character based bi-gram model with compact 8-gram and 4-gram word-level model. Another approach employed language model for final word re-ranking [5]. The top-down integration can tolerate the error in text detection or mis-recognition.

Another DNN based approach is introduced by [1], which applies a sliding window over Convolutional Neural Network (CNN) features that use a fixed-lexicon based dictionary. This is further extended in [6], through a deep architecture that allows feature sharing. In [7] the problem is addressed using a Recurrent CNN, a novel lexicon-free neural network architecture that integrates Convolutional and Recurrent Neural Networks for image based sequence recognition. Another sequence recognition approach [2] that uses LSTM with visual attention mechanism for character prediction. Although this method is lexicon-free, it includes a language model to improve the accuracy. Finally most recently, [8] introduced a CNN with connectionist temporal classification (CTC) [9] to generate the final label sequence without a sequence model such as LSTM. This approach use stacked convolutional to capture the dependencies of the input sequence. This algorithm can be integrated with either lexicon-based or lexicon-free recognition.

However, deep learning methods –either lexicon-based or lexicon-free– have drawbacks: Lexicon-based approaches need a large dictionary to perform the final recognition. Thus, their accuracy will depend on the quality and coverage of this lexicon, which makes this approach unpractical for real world applications where the domain may be different to that the system was trained on. On the other hand, lexicon-free recognition methods rely on sequence models to predict character sequences, and thus they may generate likely sentences that do not correspond to actual language words. In both cases, these techniques rely on the availability of large datasets to train and validate, which may not be always available for the target domain.

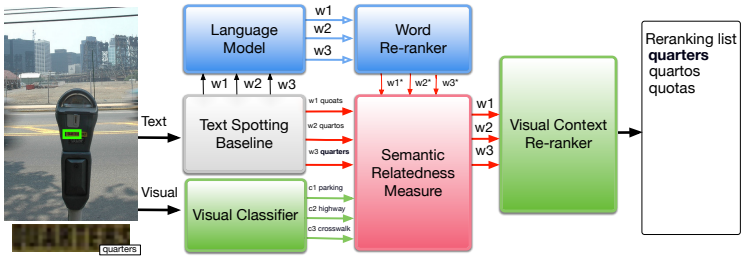

Fig. 2. Scheme of the proposed visual context information pipeline integration into the text spotting system. Our approach uses the language model and a semantic relatedness measure to re-rank the word hypothesis. The re-ranked word *quarters* is semantically related with the top ranked visual *parking*. See more examples in Figure 3.

The work of [10] also uses visual prior information to improve the text spotting task, through a new lexicon built with Latent Dirichlet Allocation (LDA) [11]. The topic modeling learns the relation between text and images. However, this approach relies on captions describing the images rather than using the main key words semantically related to the images to generate the lexicon re-ranking. Thus, the lexicon generation can be inaccurate in some cases due to the short length of captions. In this work, we consider a direct semantic relation between scene text and its visual information. Also, unlike [10] that only uses visual information over word frequency count to re-rank the most probable word, our approach combines both methods by leveraging also on a frequency count based language model.

3 General Description of our Approach

Text recognition approaches can be divided in two categories: (a) character based methods that rely on a single character classifier plus some kind of sequence modeling (e.g. n-gram models or LSTMs), and (b) lexicon-based approaches that intend to classify the image as a whole word.

In both cases, the system can be configured to predict the k most likely words given the input image. Our approach focuses on re-ranking that list using language information such as word frequencies, or semantic relatedness with objects in the image (or `visual context`) in which the text was located.

3.1 Baseline Systems

We used two different off-the-shelf baseline models: First, a CNN [1] with fixed lexicon based recognition. It uses a fixed dictionary containing around 90K word forms. Second, we considered a LSTM architecture with a visual attention model [2]. The LSTM generates the final output words as character sequences, without

relying on any lexicon. Both models are trained on a synthetic dataset [12]. The output of both models is a vector of softmax probabilities for candidate words. For each text image the baseline provides a series of k text hypotheses, that is fed to our model. Let us denote the baseline probability of any of the k most likely words $(w_j, 1 \leq j \leq k)$ produced by the baseline as follows:

$$P_{BL}(w_j) \; = \; softmax(w_j, BL) \tag{1}$$

3.2 Object Classifier

Next, we will use out-of-the-box state-of-the-art visual object classifiers to extract the image context information that will be used to re-rank candidate words according to their semantic relatedness with the context.

We considered three pre-trained CNN classifiers: ResNet [13], GoogLeNet [14] and Inception-Resnet-v2 [15]. The output of these classifiers is a 1000-dimensional vector with the probabilities of 1000 object instances. In this work we consider a threshold of most likely objects of the context predicted by the classifier. Additionally, we use a threshold to filter out the probability predictions when the object classifier is not confident enough.

3.3 Scene Classifier

Additionally, we considered a scene classifier [16] to extract scene information from each image. We used a pre-trained scene classifier *Places365-ResNet*[3] to extract scene categories.

According to the authors of *Places365-ResNet* the network achieved good result in *top-5 accuracy*, which make it ideal for multiple visual context extraction. The output from this classifier is a 365 scene categories. Also, we consider a threshold to extract most likely classes in the images, and eliminate low confidence predictions.

3.4 Semantic Similarity

We aim to re-rank the baseline output using the visual context information, i.e. the semantic relation between the candidate words and the objects in the image. We use a pre-trained visual classifier to detect objects-scenes in the image and devise a strategy to reward candidate words that are semantically related to them. As shown in the example of Figure 2 the top position of the re-ranking yields *quarters* as the most semantically related with the top position re-ranked object in the image *parking*.

Once the objects-scene in the image have been detected, we compute their semantic relatedness with the candidate words based on their word-embeddings [17]. Specifically, let us denote by $\boldsymbol{w}$ and $\boldsymbol{c}$ the word-embeddings of a candidate

[3] http://places2.csail.mit.edu/

word w and the most likely object c detected in the image. We then compute their similarity using the cosine of the embeddings:

$$sim(w, c) = \frac{\boldsymbol{w} \cdot \boldsymbol{c}}{|\boldsymbol{w}| \cdot |\boldsymbol{c}|} \tag{2}$$

4 Re-ranking Word Hypothesis

In this section we describe the different re-rankers we devised for the list of candidate words produced by the baseline DNN of Sect. 3.1.

4.1 Unigram Language Model (ULM)

The first and simpler re-ranker we introduce is based on a word Unigram Language Model (ULM). The probabilities of the unigram model computed from *Opensubtitles*[4] [18] and *Google book n-gram*[5] text corpora. The main goal of ULM is to increase the probability of the most common words proposed by the baseline.

$$P_{ULM}(w) \; = \; \frac{count(w_j)}{\sum_{w \in C} count(w)} \tag{3}$$

It is worth mentioning that the language model is very simple to build, train, and adapt to new domains, which opens the possibility of improving baseline performance for specific applications.

4.2 Semantic Relatedness with Word Embedding (SWE)

This re-ranker relies on the similarity between the candidate word and objects-scenes detected in the image. We compute (SWE) in the following steps: First, we use a threshold β to eliminate lower probabilities from the visual classifier (objects, scenes). Secondly, we compute the similarity of each visual with the candidate word. Thirdly, we take the max-highest similarity score, most semantically related, to the candidate word C_{max} as :

$$C_{max} = \underset{\substack{c_i \in Image \\ P(c_i) \geq \beta}}{\operatorname{argmax}} \; sim(w, c_i) \tag{4}$$

Finally, following [19] with confirmation assumption $p(w|c) \geqslant p(w)$, we compute the conditional probability from similarity as:

$$P_{SWE}(w|c_{max}) = P(w)^{\alpha} \qquad \text{where } \alpha = \left(\frac{1 - sim(w, c_{max})}{1 + sim(w, c_{max})}\right)^{1 - P(c_{max})} \tag{5}$$

where $P(w)$ is the probability of the word in general language (obtained from the unigram model), and $P(c_{max})$ is the probability of the most semantically

[4] https://opensubtitles.org
[5] https://books.google.com/ngrams

related context object or places to the spotted text (obtained from the visual classifier).

Note that Equation 5 already includes frequency information from the ULM, therefore it is taking into account not only the semantic relatedness, but also the word frequency information used in the ULM re-ranker above. Also, the ULM act alone in case there is no visual context information.

4.3 Estimating Relatedness from Training Data Probabilities (TDP)

A second possibility to compute semantic relatedness is to estimate it from training data. This should overcome the word embedding limitation when the candidate word and the image objects are not semantically related in general text, but are in the real world. For instance, as shown in the top-left example of Figure 3, the sports TV channel *kt* and the object *racket* have no semantic relation according to the word embedding model, but they are found paired multiple times in the training dataset, which implies they do have a relation. For this, we use training data to estimate the conditional probability $P_{TDP}(w|c)$ of a word w given that object c appears in the image:

$$P_{TDP}(w|c) = \frac{count(w,c)}{count(c)} \tag{6}$$

Where $count(w,c)$ is the number of training images where w appears as the gold standard annotation for recognized text, and the object classifier detects object c in the image. Similarly, $count(c)$ is the number of training images where the object classifier detects object c.

4.4 Semantic Relatedness with Word Embedding (revisited) (TWE)

This re-ranker builds upon a word embedding, as the SWE re-ranker above, but the embeddings are learnt from the training dataset (considering two-word "sentences": the target word and the object in the image). The embeddings can be computed from scratch, using only the training dataset information (TWE) or initialized with a general embeddings model that is then biased using the training data (TWE*).

In this case, we convert the similarity produced by the embeddings to probabilities using:

$$P_{TWE}(w|c) = \frac{\tanh(sim(w,c)) + 1}{2P(c)} \tag{7}$$

Note that this re-ranker does not take into account word frequency information as in the case of the SWE re-ranker.

5 Combining Re-rankers

Our re-ranking approach consists in taking the softmax probabilities computed by the baseline DNN and combine them with the probabilities produced by the

re-ranker methods described in Section 4. We combine them by simple multiplication, which allows us to combine any number of re-rankers in cascade. We evaluated the following combinations:

1. The baseline output is re-ranked by the unigram language model:

$$P_1(w) = P_{BL}(w) \times P_{ULM}(w) \tag{8}$$

2. The baseline output is re-ranked by the general word-embedding model (SWE). Note that this reranker also includes the ULM information.

$$P_2(w, c) = P_{BL}(w) \times P_{SWE}(w|c) \tag{9}$$

3. The baseline output is re-ranked by the relatedness estimated from the training dataset as conditional probabilities (TDP).

$$P_3(w, c) = P_{BL}(w) \times P_{TDP}(w|c) \tag{10}$$

4. The baseline output is re-ranked by the word-embedding model trained entirely on training data (TWE) or a general model tuned using the training data (TWE*):

$$P_4(w, c) = P_{BL}(w) \times P_{TWE}(w|c) \tag{11}$$

5. We also apply SWE and TDP re-rankers combined:

$$P_5(w, c) = P_{BL}(w) \times P_{SWE}(w|c) \times P_{TDP}(w|c) \tag{12}$$

6. The combination of TDP and TWE:

$$P_6(w, c) = P_{BL}(w) \times P_{TDP}(w|c) \times P_{TWE}(w|c) \tag{13}$$

7. Finally, we combine all re-rankers together:

$$P_7(w, c) = P_{BL}(w) \times P_{SWE}(w|c) \times P_{TDP}(w|c) \times P_{TWE}(w|c) \tag{14}$$

6 Experiments and Results

In this section we evaluate the performance of the proposed approaches in the **ICDAR-2017-Task3 (end-to-end)** [20] dataset. This dataset is based on Microsoft COCO [21] (Common Objects in Context), which consists of 63,686 images, and 173,589 text instances (annotations of the images). COCO-Text was not collected with text recognition in mind, therefore, not all images contain textual annotations. The *ICDAR-2017 Task3* aims for end-to-end text spotting (i.e. both detection and recognition). Thus, this dataset includes whole images, and the texts in them may appear rotated, distorted, or partially occluded. Since we focus only on text recognition, we use the ground truth detection as a golden detector to extract the bounding boxes from the full image. The dataset consists of 43,686 full images with 145,859 text instances, and for training 10,000 images and 27,550 instances for validation. We evaluate our approach on a subset [6] of the validation containing 10,000 images with associated bounding boxes.

[6] https://github.com/ahmedssabir/dataset/

6.1 Preliminaries

For evaluation, we used a more restrictive protocol than the standard proposed by [22] and adopted in most state-of-the-art benchmarks, which does not consider words with less than three characters or with non-alphanumerical characters. This protocol was introduced to overcome the false positives on short words that most current state-of-the-art struggle with, including our Baselines. However, we overcome this limitation by introducing the language model re-ranker. Thus, we consider all cases in the dataset, and words with less than three characters are also evaluated.

In all cases, we use two pre-trained deep models, CNN [1] and LSTM [2] as a baseline (BL) to extract the initial list of word hypotheses. Since these BLs need to be fed with the cropped words, when evaluating on the ICDAR-2017-Task3 dataset we will use the ground truth bounding boxes of the words.

6.2 Experiments with Language Model

As a proof of concept, we trained our unigram language model on two different copora. The first ULM was trained on *Opensubtitles*, a large database of subtitles for movies containing around 3 million word types, including numbers and other alphanumeric combinations that make it well suited for our task. Secondly, we trained another model with *Google book n-gram*, that contains 5 million word types from American-British literature books. However, since the test dataset contains numbers, the accuracy was lower than that obtained using the *Opensubtitles* corpus. We also evaluate a model trained on the union of both corpora, that contains around 7 million word types.

In this experiment, we extract the $k = 2, \ldots, 9$ most likely words –and their probabilities– from the baselines. Although the sequential nature of the LSTM baseline captures a character-based language model, our post-process uses word-level probabilities to re-rank the word as a whole. Note that since our baselines work on cropped words, we do not evaluate the whole end-to-end but only the influence of adding external knowledge.

The first baseline is a CNN [1] with fixed-lexicon recognition, which is not able to recognize any word outside its dictionary. The results are reported in Table 1. We present three different accuracy metrics: 1) *full* columns correspond to the accuracy on the whole dataset, while 2) *dictionary* columns correspond to the accuracy over the solvable cases (i.e. those where the target word is among the 90K-words of the CNN dictionary, which correspond to 43.3% of the whole dataset), and finally 3) *list* shows the accuracy over the cases where the right word was in the k-best list output by the baseline. We also provide the results using different numbers of k-best candidates. Table 1 top row shows the performance of the CNN baseline, and the second row reports the influence of the ULM. The best result is obtained with $k = 3$, which improved the baseline model in 0.9%, up to 22% *full*, and 2.7%, up to 61.3% *dictionary*.

The second baseline we consider is an LSTM [2] with visual soft-attention mechanism, performing unconstrained text recognition without relying on a lexicon. The first row in Table 2 reports the LSTM baseline result on this dataset,

Table 1. Results of re-ranking the k-best ($k = 2 \ldots 9$) hypotheses of the CNN baseline on ICDAR-2017-Task3 dataset (%)

Model	$k=2$			$k=3$			$k=5$			$k=9$		
	full	dict	list	full	dict	list	full	dict	list	full	dict	list
CNN baseline$_1$	*full: 21.1 dictionary: 58.6*											
CNN+ULM$_{7M}$	21.8	60.6	90.0	22.0	61.3	84.2	21.6	60.1	77.7	21.0	58.5	68.7
CNN+SWE$_{object}$	22.3	62.1	92.3	22.6	63.0	86.5	22.8	63.4	81.9	22.6	62.9	73.9
CNN+SWE$_{place}$	22.1	61.4	91.2	22.5	62.5	85.8	22.6	62.6	80.8	22.6	62.8	73.8
CNN+TDP	22.2	61.7	91.6	22.7	63.3	86.9	22.7	63.2	81.6	22.6	62.8	73.8
CNN+SWE$_{object}$+TDP	22.4	62.2	92.4	22.9	63.6	87.4	**23.0**	**64.0**	**82.6**	22.9	63.7	74.8
CNN+SWE$_{place}$+TDP	22.1	61.6	91.5	22.6	62.7	86.1	22.8	63.4	81.9	22.8	63.4	74.5
CNN+TWE	22.3	61.9	92.0	22.6	62.9	86.4	22.6	62.8	81.1	22.7	63.0	74.0
CNN+TDP+TWE*	22.3	62.1	92.3	22.8	63.4	87.0	22.9	63.8	82.4	**23.0**	**64.0**	**75.2**
CNN+All$_{object}$	22.3	62.1	92.3	22.7	63.2	86.7	22.9	63.6	82.1	22.7	63.3	74.3
CNN+All$_{place}$	22.2	61.8	91.9	22.7	63.1	86.6	22.8	63.4	81.9	22.6	63.0	74.0

and the second row shows the results after the ULM re-ranking. The best results are obtained by considering $k = 3$ which improves the baseline in 0.7%, from 18.72% to 19.42%.

In summary, the lexicon-based baseline CNN performs better than the unconstrained approach LSTM, since the character sequences prediction generation that may lead up to random words, which the ULM may be unable to re-rank.

6.3 Experiments with Visual Context Information

The main contribution of this paper consists in re-ranking the k most likely hypotheses candidate word using the visual context information. Thus, we use ICDAR-2017-Task3 dataset to evaluate our approach, re-ranking the baseline output using the semantic relation between the spotted text in the image and its visual context. As in the language model experiment, we used ground-truth bounding boxes as input to the BL. However, in this case, the whole image is used as input to the visual classifier.

In order to extract the visual context information we considered two different pre-trained state-of-the-art visual classifiers: object and scene classifiers. For image classification we rely on three pre-traind network: ResNet [13], GoogLeNet [14] and Inception-ResNet-v2 [15], all of them able to detect pre-defined list of 1,000 object classes. However, for testing we considered only Inception-ResNet-v2 due to better *top-5 accuracy*. For scene classification we use places classifier *Place365-ResNet152* [16] that able to detect 365 scene categories.

Although the visual classifiers use a softmax to produces only one probable object hypotheses per image, we use *threshold* to extract a number of object-

Table 2. Results of re-ranking the k-best ($k = 2 \ldots 9$) hypotheses of the LSTM baseline on ICDAR-2017-Task3 dataset (%)

Model	$k=2$		$k=3$		$k=5$		$k=9$	
	full	list	full	list	full	list	full	list
LSTM baseline$_2$	*18.7*							
LSTM+ULM$_{7M}$	19.3	79.7	19.4	74.2	19.1	68.3	18.7	60.5
LSTM+SWE$_{object}$	19.3	80.0	19.8	75.5	20.0	71.8	20.1	65.8
LSTM+SWE$_{place}$	19.3	79.8	19.7	75.3	20.1	72.4	20.0	65.3
LSTM+TDP	19.0	78.7	19.3	73.7	19.5	70.1	20.0	65.4
LSTM+SWE$_{object}$+TDP	19.4	80.5	20.0	76.4	20.3	73.1	20.6	67.2
LSTM+SWE$_{place}$+TDP	19.4	80.4	19.9	75.9	20.3	72.9	20.4	66.6
LSTM+TWE	19.5	80.6	20.0	76.2	20.1	72.4	20.3	66.4
LSTM+TDP+TWE*	19.5	80.8	20.0	76.5	20.3	73.1	**20.8**	**68.0**
LSTM+All$_{object}$	19.4	80.2	19.8	75.6	20.3	72.9	20.4	66.8
LSTM+All$_{place}$	19.4	80.1	20.0	76.2	20.3	72.8	20.3	66.5

scene hypotheses, and eliminate low-confidence results. Then, we compute the semantic relatedness for each object-scene hypotheses with the spotted text. Finally, we take the most related visual context.

In this experiment we re-rank the baseline k-best hypotheses based on their relatedness with the objects in the image. We try two approaches for that: 1) semantic similarity computed using word embeddings [17] and 2) correlation based on co-ocurrence of text and image object in the training data.

First, we re-rank the words based on their word embedding: semantic relatedness with multiple visual context from general text : 1) object (SWE$_{object}$) and 2) scene (SWE$_{place}$). For instance, the top-right example in Figure 3 shows that the strong semantic similarity between scene information *parking* and *pay* re-ranked that word from 3rd to 1st position. We tested three pre-trained models trained on general text as baseline 1) word2vec model with 100 billion tokens 2) glove model with 840 billion tokens [23] and 3) fastText with 600 billion tokens [24]. However, we adopt glove as baseline, due to a better similarity score.

Secondly, we use the training data to compute the conditional probabilities between text image and object in the image happen together (TDP). We also combined both relatedness measures as described in Equation 12, obtaining a higher accuracy improvement on both baselines, as can be seen in Table 1 and 2, (SWE+TDP) boosted the accuracy for both baseline. The LSTM accuracy improved up to 1.9% . In other hand, the CNN, with 90k fixed lexicon, accuracy is boosted up to 1.9% on *full* dataset and 5.4% *dictionary*. For example, as shown in Figure 3 top-left example, text image *kt* (sport channel) happens often with visual context *racket*, something that can not be captured by general word embedding models. Also, scene classifier SWE$_{place}$+TDP boost the baseline 1.7%

Table 3. Examples of $P(word|object)$ for each re-ranker. TDP and TWE capture relevant information to improve the baseline for pairs word-object/scene that appear in the training dataset. The TPD overcome word-embedding limitation in samples happen in training datasets.

Word	Visual	SWE	TDP	TWE	TWE*
delta	airliner	0.0028	**0.0398**	0.0003	0.00029
kt	racket	0.0004	**0.0187**	0.0002	0.00006
plate	moving	0.0129	0.00050	**0.326**	0.00098
way	street	0.1740	0.02165	**0.177**	0.17493

full and 4.8% *dictionary*. The scene classifier SWE_{place} perform better than the object classifier in instance outdoor. For instance, the spotted text in a signboard *way* is more semantically related with *downtown* than a man holding *an umbrella* in the image.

Finally, we trained a word embedding model using the training dataset (TWE). Due to the dataset is too small, we train skip-gram model with one window, and without any word filtering. In addition, we initialized the model weight with the baseline (SWE) that trained on general text, we call it TWE*. The result is 300-dimension vector for about 10K words. Also, we initialized the weight randomly but when we combined the re-rankers the pre-trained initialized model is slightly better. The result in both Table 1 and 2 button two rows shows that (TWE) outperform the accuracy of SWE model that trained on general text.

The result in Table 1 CNN shows that the combination model TDP+TWE also significantly boost the accuracy up to 5.4% *dictionary* and 1.9% *all*. Also, in Table 2, the second baseline LSTM accuracy boosted up to 2.1%. Not to mention that TDP+TWE model only rely on the visual context information, computed by Equation 7.

7 Discussion

The visual context information re-ranks potential candidate words based on the semantic relatedness with its visual information (SWE). However, there are some cases when there is no direct semantic correlation between the visual context and the potential word. Thus we proposed TDP to address this limitation by learning correlations from the training dataset. However, there are still cases unseen in the training dataset, for instance, as shown in Figure 3 bottom-left text image *copyrighting* and its visual context *ski slop, snowfield* have neither semantic correlation nor were seen in the training dataset. There are also cases where is no relation at all, as in Figure 3 the brand name *zara* and the visual context *crosswalk* or *plaza*.

The results we have presented show that our approach is a simple way to boost accuracy of text recognition deep learning models, or to adapt them to

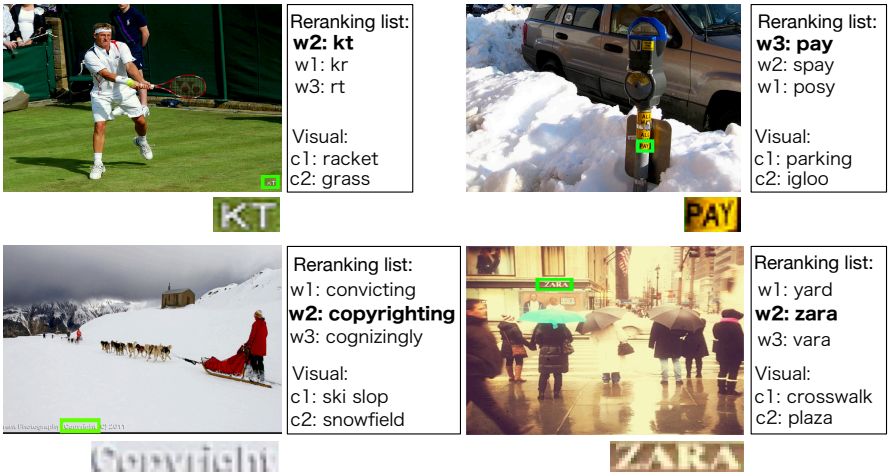

Fig. 3. Some examples of visual context re-ranker. The top-two examples are successful results of the visual context re-ranker. The top-left example is a re-ranking result based on the relation between text and its visuals happen together in the training dataset. The top-right example is a re-ranking result based on semantic relatedness between the text image and its visual. The two cases in the bottom are examples of words either have no semantic correlation with the visual or exist in the training dataset. Not to mention that the top ranking visual c_1 is the most semantically related visual context to the spotted text. (Bold font words indicate the ground truth)

particular domains, with a very low re-training/tuning cost. The proposed post-processing approach can be used as a drop-in complement for any text-spotting algorithm (either deep-learning based or not) that outputs a ranking of word hypotheses. In addition, our approach overcomes some of the limitations that current state-of-the-art deep model struggle to solve in complex background text spotting scenarios, such as short words.

One limitation of this approach is that when the language model re-ranker is strong, the visual context re-ranker is unable to re-rank the correct candidate word. For instance, the word *ohh* has a large frequency count in general text. This problem can be tackled by adjusting the weight of uncommon short words in the language model.

8 Conclusion

In this paper we have proposed a simple post-processing approach, a hypothesis re-ranker based on visual context information, to improve the accuracy of any pre-trained text spotting system. We also show that by integrating a language model re-ranker as a prior to the visual re-ranker, the performance of the visual context re-ranker can be improved. We have shown that the accuracy of

two state-of-the-art deep network architectures, a lexicon-based and lexicon-free recognition, can be boosted up to 2 percentage-points on standard benchmarks. In the future work, we plan to explore end-to-end based fusion schemes that can automatically discover more proper priors in one shot deep model fusion architecture.

References

1. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. International Journal of Computer Vision (2016)
2. Ghosh, S.K., Valveny, E., Bagdanov, A.D.: Visual attention models for scene text recognition. arXiv preprint arXiv:1706.01487 (2017)
3. Neumann, L., Matas, J.: A method for text localization and recognition in real-world images. In: Asian Conference on Computer Vision, Springer (2010)
4. Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: Photoocr: Reading text in uncontrolled conditions. In: Proceedings of the IEEE International Conference on Computer Vision. (2013)
5. Mishra, A., Alahari, K., Jawahar, C.: Top-down and bottom-up cues for scene text recognition. In: CVPR-IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2012)
6. Jaderberg, M., Vedaldi, A., Zisserman, A.: Deep features for text spotting. In: ECCV. (2014)
7. Shi, B., Bai, X., Yao, C.: 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 (2016)
8. Gao, Y., Chen, Y., Wang, J., Lu, H.: Reading scene text with attention convolutional sequence modeling. arXiv preprint arXiv:1709.04303 (2017)
9. 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, ACM (2006)
10. Patel, Y., Gomez, L., Rusinol, M., Karatzas, D.: Dynamic lexicon generation for natural scene images. In: European Conference on Computer Vision, Springer (2016)
11. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research (2003)
12. 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)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016)
14. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015)
15. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI. (2017)

16. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. (2013)
18. Tiedemann, J.: News from opus-a collection of multilingual parallel corpora with tools and interfaces. In: Recent advances in natural language processing. (2009)
19. Blok, S., Medin, D., Osherson, D.: Probability from similarity. In: AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning. (2003)
20. Veit, A., Matera, T., Neumann, L., Matas, J., Belongie, S.: Coco-text: Dataset and benchmark for text detection and recognition in natural images. arXiv preprint arXiv:1601.07140 (2016)
21. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision, Springer (2014)
22. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE (2011)
23. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). (2014)
24. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics (2017)