Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

Text Extraction from Image Data

Jindřich Libovický

Ph.D. Thesis Proposal

Supervisor: RNDr. Pavel Pecina, Ph.D.
Institute of Formal and Applied Linguistics (ÚFAL)
Faculty of Mathematics and Physics
Charles University in Prague
Malostranské náměstí 25, 180 00 Praha 1
Czech Republic

Thesis commitee: doc. Ing. Zdeněk Žabokrtský, Ph.D.
prof. RNDr. Jan Hajič, Dr.
prof. Ing. Josef Psutka, CSc.
RNDr. Pavel Pecina, Ph.D.
Ing. Mgr. Filip Jurčíček, Ph.D.
prof. Ing. Jiří Matas, Ph.D.

Prague, 2015

Contents

Introduction

A lot of people like to say that the world is overwhelmed with information that is still harder and harder to deal with, both for individual humans living in the overwhelmed world and for the technology they use. Popularity of mobile devices equipped with cameras has influenced peoples' lives in many ways recently. One of these changes is that people started to take photos as notes about things which are not of visual nature as opening hours or traffic schedules. Taking a picture of the signs became a very convenient way of storing such information, however later retrieval of such "photographic notes" with any meta-data may became very time consuming.

This is not the only case where the uses of images grows. Although the World Wide Web was original intended on to contain purely textual information (Berners-Lee and Cailliau, 1990), these days the web is full both scene images and digital-born images with text. No matter what the origin of the images is, not knowing the content of images makes automatic processing of the content of the web more difficult. By processing the purely textual content only, a lot of information is lost. Text together with the images and other multimedia content form a complex communicate which uses a semiotic code of its own kind (Warschauer and Grimes, 2007) where both the text and images contribute to both the semantics and pragmatics of the communicates.

Web applications like Picasa[1], Flickr[2] or more recently Instagram[3] publish databases of user generated scene images. However, there is no straightforward way of retrieving the images without the users providing some meta-information. The most common way of retrieving the images is to utilize manually assigned tags. The social aspect of the webs motivate the users to assign correct tags to get a feedback for their images. Assigning and processing the tags received a bigger attention leading to terms as *tagsonomy* and *folksonomy* (Voss, 2007) – a space of tags which has its own interesting semantics and pragmatics. Nevertheless, this relies on the user manually assigning the meaning which is not always the most reliable way and requires inventing extra psychological motivations for users to do so (Ames and Naaman, 2007). Retrieving images can also be based on the surrounding text, but not all the images are placed in a textual context.

Google Street View[4], a web service in Google Maps that provides panoramic views from positions along many streets in the world, is in fact a tremendously huge collection of scene images. A lot of semantic information can be assigned to the images based on the information available from maps, however having transcription of the signs on the streets, would lead to much better use of the information that is latently available. For instance knowing what is written on a traffic sign could be used for better urban navigation. ReCaptcha started to use Street View images for the manual transcription of some of the texts (Perez, 2012) which can later be used as training data for automatic transcription.

[1] http://picasa.google.com/
[2] http://www.flickr.com
[3] http://instagram.com/
[4] https://www.google.com/maps/views/streetview

These were few motivational usecases where knowing how to extract text from images may be useful. The machine vision community calls this problem Scene Text Recognition (STR) and is interested in its solution for many years. Since 2003 there has been several challenges at the ICDAR conference in text localization and recognition (Lucas et al., 2005; Lucas, 2005; Shahab et al., 2011; Karatzas et al., 2013) showing constant improvement. The system in 2003(Lucas et al., 2005) reported the localization F1-score 0.50, whereas in 2013(Karatzas et al., 2013) the best system reported F1 score 0.87. Anyway, these numbers are not directly comparable, because a slightly different dataset and evaluation protocol has been used since 2011.

Almost none of the state-of-the-are methods in STR use any external language knowledge, although there are obviously cases where knowing the language could be very useful. Images can be noisy, there can be obstacles between the observer and the text but for the human reader who the language the is in, it does not mean any major difficulties because the expectations connected the language knowledge allow them to read the text correctly. Bringing this natural ability of human readers should be the main topic is this thesis.

Firstly, we would like to incorporate the language knowledge to a STR algorithms(Neumann and Matas, 2012, 2013) and try to improve its precision this way. Secondly, we would like to dive deeper in the problem of proper reading the words that has been detected in an image. There might be more texts that do not belong to each other, and also it is necessary to find out in which order the words should be read. This step is required for the scene text to be later used in Natural Language Processing (NLP) applications as machine translation or information retrieval.

The rest of the text is structured as follows. Chapter 1 summarizes text reading from the communication theory up to detailed overview of STR algorithms and their application. Chapter 2 introduces traditional bottom-up approach to STR and focuses only on the later phase when the actual text is decoded from visual preprocessing. Some experiments conducted so far are described in the chapter. The following chapter provides a sketch of our future work in the area of embedding the scene text in visual form and its transcriptions into one common real-valued space. The logical structure of the experimental chapter is order according to how much structure is brought to the learning by the program designer and how much is left on the algorithm itself.

1. Visual World Inhabited by Text

This chapter provides an overview of the STR field starting from a very general and also very shallow ideas about scene text as a visual semiotic code and its role in human communication. Later, more technical parts follow where the STR is introduced as an artificial intelligence task with an overview of the stat-of-the-art methods. Because the dominant paradigm in this field is machine-learning approach, we also introduce the available datasets and methods of evaluation. This is followed by a overview of existing STR applications. The chapter is concluded by a non-technical reflection on the problem.

1.1 Communication Theory and Social Background

1.1.1 Images in Human Communication

Although visual communication is as old as the humankind itself, new technology introduced new ways of visual communication that very much affect the way people in the western culture communicate.

As mentioned previously in the introduction, the scene images are an important part of the semiotic codes used in mass communication. Photographs are routine part of both printed press and Internet presentations. Television and other means of distributing video are based on combining a visual, usually real scene, images with a sound track. By massive use of social networks, sending and sharing images became also a means of interpersonal and group communication. Social scientist developed a term *produsage* (Bruns, 2007) – usage by production – which characterizes how people use social media, and large proportion of this user generated content are scene images.

In 2013, 4,000 photos were uploaded to *Facebook*[1] by 1.23 billion monthly active users worldwide (Facebook, Inc., 2013). The web and mobile applications *Snapchat* and *Instagram* based on sharing images received a lot of popularity recently. According to *Instagram* official statistics[2] more than 60 million photos is shared daily by total 200 million users. Whereas the two previously mentioned applications are usually used for archiving images with a possibility of their later retrieval, the *Snapchat* is very different. Users send images to a limited list of recipients and the images are removed after a period of time given by the user both from the users' devices and the network. This of course means that the images are not available for later processing, however it nicely shows the paradigmatic shift in communication – from texting, through sharing images to the moment when an image itself can be used as a primary unit of communication. This phenomenon of course brings attention of social scientists and has been recently studied e.g. by Oeldorf-Hirsch and Sundar (2010), Litt and Hargittai (2014), Panahi et al. (2012) or Poltash (2013).

According to Marshall McLuhan's periodization of the human history (McLuhan, 1962), the advent of the electronic media (radio and television) in the middle of the 20th century meant an end of the so-called Gutenberg's era and moved people closer to supposedly more natural state of communication which is based on speech

[1] http://www.facebook.com

[2] http://instagram.com/press/, accessed 18/8/2014

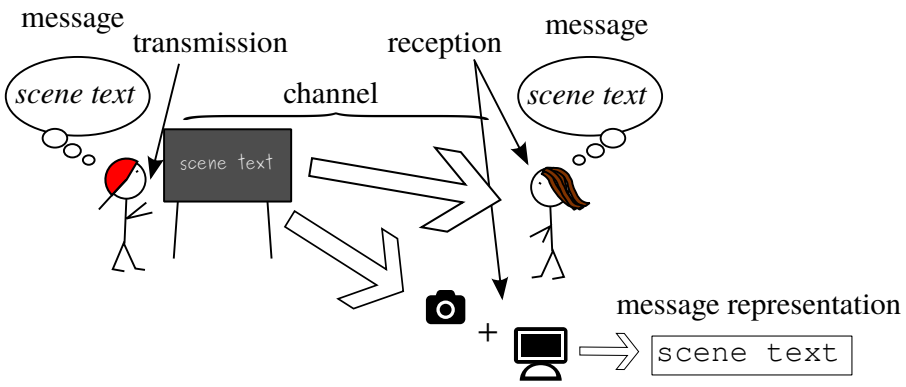

Figure 1.1: An illustrative scheme of reading scene text as a communication in Shannon's model

and shared visual experience – basically showing scenes to each other. He sees a medieval village as an ideal form of people living together and argues that electronic media (the Internet did not exist that time) help to build a unified *global village*. His ideas were applied also to the online communication with emerging *collective intelligence* (Lévy and Bonomo, 1999).

Nevertheless, some thinkers do not agree with this view. For instance, Postman (2006) accepts McLuhan's periodization, but is rather critical to this shift in human communication. He partially agrees with the critical theory school (Geuss, 1981) and argues that showing scenes and using a complex visual semiotic code (with the scene text as a part of this code) instead of formulating thoughts in a paragraphed text leads the society to a shallow thinking.

Not only the standard sings and placard form a semiotic code of its own kind, we can consider digital sending and sharing images as a relatively new way of communication with its own semantics and pragmatics. Scenes forms a complex semiotic code, the scene images very frequently contain some text. Sometimes, texts on an image often have nothing in common with what the message of the image is, but sometimes the scene text is often a prominent part of the message the sender tries to send by the image and is crucial for understanding the scene. Except straightforward applications mentioned in the introduction, if we would ever want to attempt to computationally process such code, STR is an important step for doing so.

1.1.2 Text Recognition as a Communication Process

In this context, we implicitly understand reading of a text as part of a linear communication process in natural language. Using the widely accepted Shanon and Weaver model of communication (Craig, 1999; Shannon, 1948), the senders first encodes the message they want to transmit into a natural language and then encode into the graphemes that form the actual text. The message is then transmitted via a channel which is in this case the medium on which the text is written, and the light carrying the information to the receivers' eyes. The channel can be of course noisy – there might be some obstacles for the light, the medium can be corrupted. The receivers, very likely, decode the graphemes that were transmitted jointly with understanding the message that senders wanted to send.

While doing the automatic text recognition, not only the recipient, but also the communication channel is different. The receivers' eyes are replaced by a digital camera that captures the light in a similar way we believe human eyes do. Instead of being interpreted by a human, the image is discretized into a set of pixels withs discrete numeric qualities. An output of this channel is then used in the decoding procedure which ends with the graphemes more or less successfully localized and recognized. Note that in this case, the graphemes are not retrieved jointly with understanding the text. In fact, they are recognized without any understanding at all. Nevertheless, the human expectation originating from their understanding can be hopefully simulated by language modeling.

Of course, the machine recognition process does not necessarily need to end at the moment the graphemes are recognized. Grouping the recognized tokens into larger logical groups and labeling the groups to some semantic categories can bring us closer to machine understanding of the textual content of the images.

Whenever people or devices communicate, the complexity of decoding a received message strongly depends on a protocol the communicating parties agreed on before. For example, if all people in a room know that when they hear a whistle they should leave a room and gather in front of a building, just one bit of information is sufficient to trigger a very complex situation. On the other hand, if all the people were explicitly said what to do at the moment of the whistle, much more information would be transmitted. The protocol that is used in case of scene text is indeed the natural language.

The fact that people can read text very quickly and very robustly can be attributed to the knowledge of the "protocol" on all levels of the language structure, starting with pragmatics and ending with phonology. Expectancy of word meanings and text structure inspired the famous Internet 2003 mem claiming that people can read words with permuted characters without any difficulties (Rayner et al., 2006). On the other hand, if we meet an unknown word and heuristics based on language pragmatics fail, we can easily read it using morphology and phonology of the language the word is from because we also have some expectations on that level. This can be supported by studies showing that reading permuted and not-permuted words are different mental processes (Rashid et al., 2010). Ultimately, if we know the alphabet, we can read any strings in any languages, even though we may not be able to pronounce it.

1.2 Scene Text Recognition

STR is a subfiled of computer vision whose goal is to develop techniques of automatic acquisition of text depicted on photographs taken in the natural environment. It is sometimes also called "Scene Text Understanding" (Mancas-Thillou and Gosselin, 2007). It is related to the well known problem of Optical Character Recognition (OCR) which has been studied for a very long time. (Schantz, 1982; Mori et al., 1992).

Its research started in 1950's and it was considered to be a solved problem in 1990's (Govindan and Shivaprasad, 1990; Øivind Due Trier et al., 1996). Because of the very good performance of the OCR algorithms (the last 1996 competition reported character accuracy between 96 % and almost 100 %, depending on the type of text (Rice et al., 1996)) we can usually take the OCR outputs as if it would be a digital-born text and use the standard NLP algorithms for the further processing.

The classical OCR can rely on properties of the printed text. It is usually printed in straight lines with only few characters exceeding the line height. The colors of the characters and of the background do not change much and they are usually very contrasting. There are also no shadows. None of these properties does not hold for the scene text.

1.2.1 Text Detection and Localization

Zhang et al. (2013) recently brought the comprehensive survey of the STR techniques covering publications from 2000 to 2013. They stress few dominant scene text features that the STR algorithms need to take in account. These are:

1. the size of the characters can differ;
2. the directions of the text can differ and geometric distortions can occur;
3. characters of the same segment tend to have similar colors;
4. scene text is usually design to be easily read – it is contrasting with clear boundaries;
5. perspective distortions can dramatically effect the recognition.

They also divide the STR pipeline into three steps: text detection and localization; text enhancement and segmentation; and text recognition (OCR). However, this division does not accurately fit for all of the cases.

There are few basic approaches that can be taken in the localization and detection part. Except the naive sliding window approach, the detection is mostly:

- *edge based* – It is the historically oldest method. An edge detector (Canny, 1986) is used to extract features. It is easy to use, however it easily influenced by shadows and highlights. It was used e.g. by Liu and Samarabandu (2006), Ye et al. (2007) or Ou et al. (2004).
- *texture based* – It uses the assumption that the text has different texture than the background. The approaches use methods like Gaussian filtering, Wavelet decomposition or Fourier Transform. It is used e.g. by Zhou et al. (2011) or Pan et al. (2010).
- *connected components* – It is a bottom-up approach that is based on gradual grouping smaller components into bigger ones. Algorithmically, it can be done using various methods. Conditional Random Fields (CRFs) can be used to classify super-pixels (Pan et al., 2010, 2011), or hierarchical clustering based on various features can be used (Wang and Kangas, 2001).
- *based on extremal regions* – Characters are spotted as extremal regions (Matas et al., 2004) in some color space projections of the image. It was used by e.g. by Neumann and Matas (2012, 2013), Chen et al. (2011) and by the so far best localizer by Yin et al. (2013).

- *stroke based* – A text is modeled as a combination of strokes of different orientation of almost constant stoke width. After publication of the Stroke-width transform (Yao et al., 2012), Zhang et al. (2013) considers methods using strokes to be the most promising. This method was used e.g. by Epshtein et al. (2010), Srivastav and Kumar (2008) or Yi and Tian (2011).

Combinations of these approaches are also used (Pan et al., 2009; Lee et al., 2010; Minetto et al., 2010). For other approaches and more details we refer the reader to Zhang et al. (2013).

After the great success of a Convolutional Neural Network (CNN) for image classification (Krizhevsky et al., 2012), the deep learning techniques became very popular, spread to all fields that use machine learning. Jaderberg et al. (2014b) used deep CNN for text localization. Neural networks has been also used for the cropped word recognition which is described later in the following section.

1.2.2 Localized Text Recognition

After the text is detected and localized, further processing follows to receive the actual transcription of the localized text. There are basically two options how this step can be done. The historically older approach is to remove the perspective distortions and to binarize the image (to have black text and white background), such that it would be possible to transcribe it using a standard OCR program. Another option is to create a model that recognizes the text as is, either monolithic (as a Neural Network (NN)) or with a pipeline using explicit character segmentation.

The *text extraction and enhancement* phase is supposed to make the detected text ready to be processed with a classical OCR program as commercially available ABBY[3], Omnipage[4] or open source Tesseract[5]. It usually means text binarization – process which produces black letters on white background. However, some experiments show that better results are achieved using classifiers directly trained on segments (Wang and Belongie, 2010) of the image without binarization.

There is also research focusing on the later parts of the pipeline. Assuming the detection and localization have been done perfectly, they focus only on the recognition. Recent approaches usually do not perform the image binarization and try classify the localized characters directly without using an OCR program. A comprehensive study by Campo et al. (2009) suggests the Histogram Oriented Gradient features (Dalal and Triggs, 2005) and the nearest neighbor classifier as the most suitable method for natural scene character detection.

Wang and Belongie (2010) focus more on text spotting. They first detect the possible characters using sliding window and construct a graph of all potentially adjacent character hypotheses. Then, they find a path in the graph representing the word which maximizes an objective function. The objective function tries to maximize the classification score of the image windows and local geometrical similarity of neighboring characters.

Another possibility is to recognize only words from a given lexicon. Mastering this task could be sufficient for applications like navigation for visually impaired

[3] http://www.abby.com
[4] http://www.nuance.co.uk/OmniPage-18
[5] http://code.google.com/p/tesseract-ocr/

people or analysis of additional text on traffic signs. The method of Wang et al. (2011) uses a *lexicon* (a list of 50 to 500 words) for each processed image and words in the image. It works similarly as the previous word spotting algorithm Wang and Belongie (2010), but it adds a classifier that decides whether the spotted word is present in the image. The method achieves high precision in text recognition, but its applicability is limited by the requirement of having a small fixed lexicon for each image prior to the recognition. Therefore it cannot be used to acquire new textual data because the output method is limited by the words in the lexicon. Similar approach is taken by Mishra et al. (2013) in the image retrieval system which is discussed later (see Section 1.3).

Mishra et al. (2012b) and Novikova et al. (2012) were able to outperform these methods (Wang et al., 2011) using probabilistic graphical model with priors given by a dictionary to word recognition. Whereas Mishra et al. (2012b) uses a CRF for the segmentation decoding only (while using the first-best recognition hypotheses for the segments), Novikova et al. (2012) jointly decodes both segmentation and character recognition in a probabilistic graphical model which reduce to a weighted finite state transducer.

TextSpotter (Neumann and Matas, 2012, 2013) uses a lexicon-free method for STR which operates in an end-to-end setup, i.e. it both text localization and recognition stages and it requires no manual annotation or lexicon to transcribe text in images. It first detects image regions corresponding to individual characters, joins them into text line hypotheses, and then recognizes the characters using an OCR classifier.

The only linguistically motivated work in this area was the dissertation of Field (2014). She introduces a syllable based language model. She assumed a lot of words appearing in scene are not dictionary words, however people immediately know how to read them, therefore can be parsed into syllables. A CKY parser is used for decoding where the terminals' score are given by an OCR classifier and non-terminals are during the syllable parser training.

She also developed another technique that attempts to overcome normal spell checking dictionary sparsity. She used frequency of unigrams in the Google n-gram corpus to rescore the word hypotheses. This approach assumes that the non-dictionary words are usually proper names which are likely to be observed on the web and can help to discriminate random string.

Recently, two approaches based on deep learning outperformed the previous methods. Su and Lu (2014) adapted a method known from handwriting recognition. They treat the word image as a sequence of columns for which they compute feature vectors. These vectors are interpreted as an observed signal similarly as in speech recognition. A recurrent NN is used for sequence labeling.

The best results so far in cropped word recognition was achieved by Jaderberg et al. (2014a). They used a CNN to classify cropped words into 90,000 classes – a dictionary words. A noteworthy property of such network is that the convolutional filters does not capture any spatial information, i.e. only bag of events as character unigrams or bigrams is used for the classification. Indeed, such learning requires huge amounts of data which the authors solved using synthetic data which we discuss more in Section 1.2.4. Unfortunately, the work does not mention whether the representation that is used in the last soft-max layer of the network is general enough

such it can be retrained with a different dictionary. We hypothesize this might by case which would even more interesting result.

1.2.3 Evaluation

STR as an artificial intelligence task is evaluated on benchmarks which we believe are a representative sample of instances that can appear when developed methods are applied in practice. Here, we intuitively rely on the induction principle because there are no formal assumptions we can rely on. Because STR is usually solved by machine learning, the datasets contain both the train and test data, so the method are comparable also from the machine-learning point of view. A comprehensive list of available datasets is presented in Section 1.2.4.

In case we deal only with the cropped word recognition task, the evaluation is relatively straightforward. Usually, accuracy as a proportion of correctly recognized words is measured which suits well for both unlimited recognition and a dictionary-based recognition. The option one would naturally consider is measuring the average edit distance on a dataset which makes sense only if arbitrary strings are allowed.

The situation is much more complicated for the localization task and end-to-end recognition. In general, precision, recall and F_1 measure of the word retrieval is computed, however the problematic moment is the definition of when two word areas match. The ICDAR evaluation protocol was slightly different for each of the competitions(Lucas et al., 2005; Lucas, 2005; Shahab et al., 2011; Karatzas et al., 2013). The development of the metric tried to minimize the possibility to optimize the algorithms with respect to the metric properties, which in many case intuitively does not mean any improvement in the localization itself. Whereas the previous competitions evaluated overlap of bonding boxes which were horizontal rectangles surrounding the words, the 2015 evaluation protocol uses general quadrilateral as the bounding boxes. It is connected with introducing the category the category of incidental scene text which are random shots from an urban environment where is no reason to assume the texts should be horizontal.

Some papers (Wang et al., 2011) made a strict distinction between words spotting and text recognition. By *text spotting* they mean finding only words from a limited dictionary in the image and evaluate it as a retrieval problem. On the other hand, the *text recognition* does not limit itself to the known words and tries to detect and possibly transcribe all text in the image. Wang et al. (2011) operated with a dictionary of size 50 – 500 words. Nevertheless, most of the recent methods operates with a dictionary and try to come with dictionary based methods that work well also for large vocabularies Roy et al. (2014). Dictionaries of various sizes are also provided with some datasets (Mishra et al., 2012a; Wang and Belongie, 2010). A kind of an extreme case is the CNN by Jaderberg et al. (2014a) which operated over a dictionary of 90,000 words. If we disregard some special cases like serial numbers and license plates, using such a large dictionary does not impose significant limits for recognition. Because of this, the ICDAR 2015 evaluation protocol includes vocabularies of three sizes with the last one being the one with 90,000 words. The 2015 evaluation protocol also introduces a concept of “not-care” words which are difficult to read. Localizing and detecting of such words is not considered neither as a correct one, nor as a false positive.

The precision and recall of end-to-end recognition is computed as a conjunction of the correct detection string correctness. If we define recognition accuracy as a proportion of correctly transcribed words in correctly detected words, we can trivially show that the precision, recall and F_1 measure for the end-to-end recognition are the values for localization multiplied by recognition accuracy. In ICDAR 2015, the evaluation is case insensitive, although it did not use to be so.

1.2.4 Available Datasets

The main technique used in STR are the supervised machine learning algorithms when the most difficult engineering part is design the features. This raises the need to have some annotated data on which the models will be trained.

The Chars74k [6] (de Campos et al., 2009) is a dataset of cropped characters for Latin and Kannada alphabet. There 64 classes (0-9, a-z, A-Z) for the Latin alphabet. Approximately one tenth is cropped from real scene images, one tenth are characters hand-drawn on a tablet and the rest are synthetically generated characters. The training data for the character recognition are not a big issue because they can be easily obtained synthetically in arbitrarily high amount.

The *IIT 5k-words database*[7] (Mishra et al., 2012a) consists of 1,120 images with 5,000 words cropped from photographs mined via Google Images. Each image is provided a dictionary of 50 words.

The most frequently used benchmark for STR is the *ICDAR Dataset* in various versions.(Lucas et al., 2005; Lucas, 2005; Shahab et al., 2011; Karatzas et al., 2013) Datasets used in 2003 – 2013 challenges consisted of almost the same images with only minor differences in the annotation scheme. There are images taken both indoors and outdoors, including traffic signs, business signs, institutional navigation signs, book covers and other types of text, without this fact being explicitly annotated. It has been published in various versions. The first version from 2003 (Lucas et al., 2005) consists of 251 images with 1,110 words with both characters and word location and labels. It was replaced by the 2011 version (Shahab et al., 2011) which contained the word locations and labels. In 2013 (Karatzas et al., 2013), some errors in annotations and duplicate images were removed. Nevertheless, the annotation is still inconsistent in capturing punctuation. It was divided into training set with 229 real-world images containing 849 words and test set with 1095 words in 233 images. All words are in English. Its major drawback was also that it contains only horizontally oriented texts.

For the 2015 challenge, a new dataset has been released[8]. Unlike the previous years when the same data were used, now two datasets were published. The dataset from the previous competitions is called "focused scene text" because the text is usually the central object in the image and usually is almost horizontal.

The second dataset is called "incidental scene text". It consists of 1,000 training and 500 test images taken in 2014 on streets and in public transport in Singapore. Because the photos were taken by camera that a person wore on his head, thee text are in various angles and under string perspective distortions. Reflections and ob-

[6] Available at http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

[7] Available at http://cvit.iiit.ac.in/projects/SceneTextUnderstanding/IIIT5K.html

[8] The data and the evaluation protocol was presented at http://rrc.cvc.uab.es/, and are available after registration. A paper summarizing the data and evaluation will be published this year.

stacles appear often. In both dataset, the text areas are annotated as general quadrilaterals. Each bounding box is also give a small vocabulary of 50 words. There are also medium-sized vocabularies for the images as well as the 90,000 vocabulary by (Jaderberg et al., 2014a).

The KAIST Scene Text Database[9] (Jung et al., 2011) consist of 3,000 images in the resolution of 640×480 pixels with texts in English and Korean. The images are divided into categories according to device used to take the images (camera, mobile phone) and the environment they were taken in (indoor, outdoor, shadow, light, night, book covers). The dataset is painstakingly annotated with word and character bounding boxes, the textual content and also with the exact coordinates of the character borders.

The *MSRA Text Detection 500 Database (MSRA-TD500)*[10] (Yao et al., 2012) consist of high resolution images with annotated bounding boxes and its transcription in Latin (English) and Chinese scripts. Only the text localization bounding boxes are annotated, not the actual textual content. The training part consist of 300 images with 1,068 bounding boxes, the test set contains 651 bounding boxes in 200 images.

The *Natural Environment OCR (NEOCR) Dataset*[11] (Nagy et al., 2012) is a set of 659 real world images with 5,238 annotated bounding boxes. The dataset could be considered to be more difficult for STR because many of the texts are very distorted. The dataset it annotated in the LabelMe scheme (Russell et al., 2008). The bounding boxes are annotated both as a minimum horizontally oriented rectangles, but also rotated rectangles fitting the boxes distortions. The text snippets are in various languages, most of them German and English or language independent as numbers or company names

The *Street View Dataset*[12] (Wang and Belongie, 2010) consist of images harvested from the Google Street view. The annotators were first ask to find particular venues on the map and then to locate and transcribe the business signs in the images. The annotation is available in the ICDAR format, but unlike the ICDAR dataset, only the business sign are annotated and other pieces of text are neglected. The training set consists of 100 images with 211 words and the test set consists of 250 images with 514 words. The images were harvested in the U.S. cities, so the dataset is in English.

reCAPTCHA (Von Ahn et al., 2008), a project for collecting data for OCR became a famous example of so called human computation(von Ahn, 2009). CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) are system that helps to prevent massive abuse of web services by forcing the users to perform a task which is easy for humans, but extremely difficult to master algorithmically. The project was developed not to waste these mental effort – words that were not recognized well from the scanned New York Times archive, were visually malformed to make worse readable. The error rate of the original OCR was almost 20 % due to the faded ink yellowed paper. People achieved 99 % accuracy and after one year of running the system over 440 million suspicious words was correctly transcribed.

[9] Available at http://www.iapr-tc11.org/mediawiki/index.php/KAIST_Scene_Text_Database

[10] Available at:http://www.iapr-tc11.org/mediawiki/index.php/MSRA_Text_Detection_500_Database_%28MSRA-TD500%29

[11] Available at www.iapr-tc11.org/mediawiki/index.php/NEOCR:_Natural_Environment_OCR_Dataset

[12] Available at http://vision.ucsd.edu/~kai/svt/

The project was acquired by Google in 2009(von Ahn and Cathcart, 2009) and is now used for project like Google Books Search and Google News Archive Search. In 2013 it started to to use also addresses extracted from the Google Street View images Perez (2012). It seems to us that the text for the transcription is likely extracted using an unsupervised technique similar to one introduced by Netzer et al. (2011).

As mentioned previously, Jaderberg et al. (2014b) successfully used synthetic data to train cropped word recognition. Although they have not published the code for generating the training data, they made the generated dataset of 9 million images publicly available. For the 90,000 words vocabulary they generated 10 examples for each of the words. The dataset does not contain any test data because it is supposed to be tested on the non-synthetic benchmarks.

1.3 Scene Text Applications

Processing of texts having origin in OCR has been a part of technological practice for many years. The advanced state of the OCR technology (see allowed development of commercially successful solutions both for institutional (smartFix Deckert et al. (2011) or Open Text Capture[13]) and personal use (Intellix Schuster et al. (2013)). Famous examples of OCR applications are Google Books[14] search with 20 million of scanned volumes(Howard) and K-NFB Reader[15], a mobile application that reads aloud for visually impaired people.

Mastering STR would enable similar applications also for scene text, most notably information retrieval and extraction and machine translation. In case of the scene text, we need to deal with a different nature of the text we are processing. It usually provides only very little textual context which can left the text itself very ambiguous, although its meaning can be clear in the visual context. For example the word "*visa*" has a different meaning (and therefore different translations to other languages) on a direction sign at an embassy and on a list of accepted credit cards in a shop.

Relatively few applications that are based on processing scene text exist, however not so many works have been published on this topic recently. Moreover, most of the research papers about the application has been published more than a decade ago.

One of the desirable applications is image retrieval by a textual query – retrieving images containing a given text could be used not only if users search for particular images, but also for navigation in captured videos. Such a prototype was designed by Thillou et al. (2005). There was also a Belgian project Sypole[16] which seems to have ended in 2007 (Gaudissart et al., 2004).

An interesting application was done by Posner et al. (2010). They built a robot whose task was to move along streets in a town while using information signs for orientation.

Mishra et al. (2013) conducted an experiment that showed that using the state-of-the-art recognition technology (Neumann and Matas, 2013) for indexing the im-

[13] http://www.opentext.com/what-we-do/products/information-exchange/capture-and-recognition/opentext-capture-center

[14] http://books.google.com

[15] http://www.knfbreader.com/

[16] http://www.tcts.fpms.ac.be/projects/sypole/?lang=en

ages leads to very poor results. They received precision of 25 % while making queries on the Street View Dataset. In the strict case, even one error in the transcription prevents the image from being found. Instead, they proposed a fuzzy search methods that treats the queries as bag of characters or bigrams which are search in images preprocessed to contain windows with probability distributions over characters. This way they were to able to increase the precision to 52 %.

Ironically, most of the application papers were published in few years after 2000 when the recognition technology was hardly usable. One of the older application are image retrieval based on textual context Smeulders et al. (2000), retrieving videos based on subtitles (Sivic and Zisserman, 2003), Chinese image retrieval (Luo et al., 2003) or machine translation of scene text (Haritaoglu, 2001).

The most famous example of application that can deal with the scene text is the mobile application *Google Goggles*[17] released in 2011(Palm and Zhang, 2011). It can be used for web search based on images taken from handheld devices. One of its features is also STR and then searching the web for the detected text. No research papers has been published about the application. Bing Translator for Windows Phones allows also translation of scene text (Dendi, 2012).

[17] http://www.google.com/mobile/goggles

2. Bottom-up String Decoding in Scene Text Recognition

String decoding is one of the steps of traditional STR pipelines. It is a process of finding the correct transcription in the moment when we have some hypotheses about the text location, positions of the particular characters and what characters actually can appear at a positions. It is a natural moment where some kind of language knowledge can be incorporated.

This is in principle a very similar problem many structured prediction problems in NLP such as decoding sentences in target language in machine translation, hypotheses decoding in automatic speech recognition or all problems which can be reduced to sequence labeling as part-of-speech tagging, named entity recognition or text chunking.

While doing this, there are two general approaches we can use: compute the globally optimal solution while using only local features; or use global features with only an approximative inference algorithm. We discuss these approaches in Section 2.5 and 2.6.

The experiments in this chapter follow our previous results which have been already published (Libovický et al., 2014).

2.1 Task Definition

We do our decoding experiments using the TextSpotter (**?**) pipeline from which we use the character localization and OCR. In TextSpotter, each character appearing in an image may be detected several times (each detection corresponds to a different image segmentation) and each such detection may be assigned one or more character labels (transcription hypotheses) by the OCR module. Such an approach is beneficial, because it allows to keep multiple hypotheses for character segmentation and their labels for a later stage of the pipeline, where the final decision can be made by exploiting context of the character in the word (we define a *line* simply as a sequence of characters containing spaces where the spaces are delimiter of what we call *words*.

For that purpose we define a transcription hypothesis graph $G = (V, E, s, t, l)$ where (V, E) is a directed acyclic graph, $s \in V$ is a start node with no incoming edges, $t \in V$ is a target node with no outgoing edges. $V = \{s, t\} \cup V_c \cup V_s \cup V_{\neg s}$ is a set of vertices where V_c is a set of character hypotheses and V_s and $V_{\neg s}$ are separator vertices. This means $V_s = \{(u, v, \texttt{space}) \mid u, v \in V_c,\ u \text{ can follow } v\}$. Function $l : V \to$ `[a-zA-Z0-9]` is label assignment for the vertices, a space for all vertices in V_s, an empty string for vertices in $V_{\neg s}$ and s and t and character recognition hypotheses for vertices in V_c. The set of edges E then connect character hypotheses with the corresponding separator vertices and the separator vertices with the following character hypotheses.

The goal of hypothesis decoding is to find a path $\mathbf{y} = (y_1 = s, y_2, \ldots, y_{k-1}, y_k = t)$ from the start node to the target node such that the concatenation of labels of the vertices on the path, $l(y_2), \ldots, l(y_{k-1})$, forms the ground truth string – a string an annotator transcribed from the image.

2.2 Related Work

In this section we summarize the methods used in the final stages of STR when the final strings are produced from the character and segmentation hypotheses. A typical solution of this problem is finding a global extreme based on local features capturing properties of neighboring characters or at most character trigrams.

Mishra et al. (2012b) use CRF to decide between different segmentations. Each bounding box is represented only by its first-best character recognition and is assigned a unary feature – its classification score. Potentially neighboring character hypotheses are connected by binary factors encoding language model scores. The feature weights are estimated empirically.

Roy et al. (2013) first detect a line on which characters lie. Then a Hidden Markov Model is used to decode the string from a set of observed in sliding windows. This approach allows both multiple segmentation and multiple hypotheses per window, however by having most of the windows emitting empty output they loose possibility to use a language model score for adjacent characters.

A similar approach to the one presented in this paper is used by Shi et al. (2013). After a text area is detected, it is segmented into character bounding boxes. For each bounding box, a classifier produces several hypotheses which are then decoded (disambiguated) using a linear chain CRF. However, in this case the character segmentation is decided beforehand and multiple transcription hypotheses a segment hypotheses are allowed.

Weinman et al. (2014) use structured Perceptron for optimization of the weights in a dynamic programming decoding. We used a similar approach with structured Perceptron and Stuctured SVM in our experiments with TextSpotter and we discuss these learning techniques later in this chapter.

In case of recognition of words from a prior lexicon, the decoding can be constructed such that it could only produce the words listed in the lexicon. Novikova et al. (2012) employ a trie-shaped weighted finite state automaton, Wang et al. (2011) use dynamic programing for searching areas that may correspond to words from a dictionary. This produces a list of candidate locations which are filtered using a binary classifier.

Unlike the previously mentioned approaches, the following can employ more complex language features. An end-to-end STR pipeline, PhotoOCR (Bissacco et al., 2013), use a longer n-gram language model in a machine-translation-like beam search to explore all possible paths with pruning over the least probable partial paths.

As we mentioned in Section 1.2.2, Field (2014) used a probabilistic context-free grammar for English syllables to find the most probable parse of the character hypotheses.

2.3 Data

We ran the first stages of the TextSpotter pipeline and generated the transcription hypotheses for hypothesis graphs.

The ICDAR 2015 dataset of incidental images was used to generate the training data for our experiments. The training set consists of 1,000 real-world images with 11,886 words in total, 7,418 of them “not care” words. For each word, the ground

truth annotation is given in the form of bounding box coordinates. The training data was generated by running the initial stages of the TextSpotter pipeline on each image to generate graphs of word transcription hypotheses and then in each graph, the ground truth path representing the correct transcription is selected (if it exists) by matching graph vertices to a manual annotation of graph areas.

This process produced a total of 1,693 line graphs (including multiple graphs for the same word because the TextSpotter pipeline processes the same image several times using different visual transformations of the original image). We were able to match a subset of 631 graphs (931 words) with the ground truth, out of which a random sample of 442 graphs was used to train the model and the remaining 189 graphs were used for intrinsic evaluation.

Figure 2.1: Examples of the lines automatically cropped from the ICDAR 2015 incidental dataset.

When we did experiments with single words decoding, we used the training part of the ICDAR 2013 dataset. We generated the graphs for the words and did the annotation of the graphs automatically – the single word annotation was provided with the dataset, moreover there were no "not care" words which were often within the lines in the 2015 dataset. This way got 812 word graphs from 229 images, from we used 568 for training and 244 for testing.

2.4 Evaluation

The evaluation scheme for decoding is very simple. We report accuracy – a proportion of correctly decoded strings and average edit distance of a decoded string from the ground truth annotation.

Unfortunately, we omit the most important part of evaluation – the extrinsic evaluation when the decoding procedure is fully integrated into the recognition pipeline. The version of TextSpotter that we used processes separately various visual transformations of the image and a heuristic rule joins the string from these so-called channels together at the end. Unfortunately, so far we have not been able to adapt this phase in coordination with the TextSpotter authors such that it would not favour wrong strings in this phase. This is still left for future work.

Nevertheless, we ran the experiments with our decoding on the ICDAR 2015 datasets and submitted it to the competition. Even if our decoding significantly

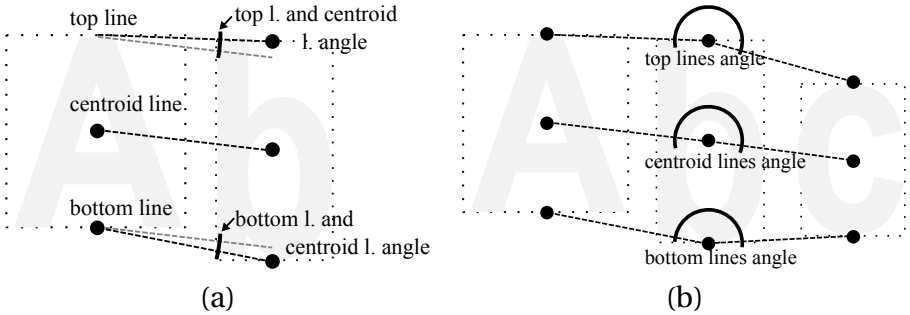

Figure 2.2: Typographical features used in the first-order model (a) and the second-order (b) model.

improved the TextSpotter's performance it would be probably still below the state-of-the art methods.

2.5 Decoding Using the Local Features

A natural way of solving the task as we defined it previously is by finding the maximum path through a hypothesis graph. It is a well known optimization task with a fast algorithm which guarantees finding a global maximum. The structured prediction which has been intensively studied in the NLP context gives us methods how to estimate a function for evaluation of each edge, such that the maximum path will likely represent the correct string. Finding features – numeric descriptors of the graph edges and nodes can be done also very naturally.

This approach has one significant drawback – the features can only capture properties of only potentially neighboring characters and does not take in account properties of the whole string. We pay this price for the possibility of easy finding the globally optimal solution. As we will see later, there is also an option to give up the requirement of having the globally optimal solution which will allow us to use longer distance features.

2.5.1 Features for Local Decoding

Our first-order (bigram) model extends the method by adding the following 20 typographical and language features:

- ratios of width, height and area of character on the edge,
- mutual angles of the top line, bottom line and centroid line (see Figure 2.2a),
- conditional character bigram probability;
- binary features encoding character patterns (two digits, lower case + upper-case letter, both the same case, lower to upper case); and
- bias for making a space and not making a space.

All the features are included in the feature vector twice, once conjoined with emitting a space and once without. All the ratios were computed as absolute values

of the difference of logarithms. All the features are standardized to have a zero mean and unit variance on the training data.

We also experimented with introducing the trigram context which would allow us to introduce a richer context for the features (see Figure 2.2 for an example which angles and distances could be used), our result in a slightly different setting **?** showed that the slight improvement does not pay off in context of computational demands due to the quadratic increase of the graph size.

2.5.2 Extremal Path Decoding

The most natural way how to do an inference in such a model is to find the correct string as a maximum path through the graph.

In TextSpotter, the model parameters (linear combination coefficients) are tuned by a simple grid search. An alternative is to treat the problem as a standard classification and train a classifier to predict for each edge how likely it is to lie on the ground truth path. Such classification is local, the edges are scored independently from each other; those lying on the ground truth paths are used as positive training examples and the others as negative ones (we resampled the training data to have the same proportion of positive and negative examples). We examined several standard machine-learning algorithms implemented in WEKA (Hall et al., 2009): Logistic Regression, Support Vector Machine (SVM) with various kernels, Random Forest, and multilayer Perceptron with various hidden layer configurations.

In local classification, the information about the final output is ignored during training. The constrains for the output structure (a graph path) apply in decoding (testing) phase only. A more appropriate solution is structured prediction, where the decoding algorithm is also used during training, the classification is global and allows the parameters to be optimized with respect to the inference algorithm. To employ the structured prediction we need to formulate the problem as finding a structure (in this case a path) which is maximal with respect to a function that is a dot product of a weight vector and feature function of the structure, here a sum of the feature values along the path. This means:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}\in\mathcal{Y}_\mathbf{x}}{\operatorname{argmax}} \mathbf{w}^T \Psi(\mathbf{x},\mathbf{y}) = \mathbf{w}^T \sum_{e\in\mathbf{y}} \phi(e)$$

where $\mathbf{w}$ is a learned m-dimensional weight vector.

While training a prediction model, we want to estimate the weight vector $\mathbf{w}$, such that maximum number of the training instances would be correctly classified. We examined two state-of-the-art techniques for the weights optimization: structured perceptron (Collins, 2002) and structured SVM approximated by the cutting plane algorithm (Joachims et al., 2009).

Structured perceptron (Collins, 2002) is a simple modification of the standard perceptron algorithm. The weight vector is iteratively updated by the difference of the feature vector of the currently estimated solution and the ground truth solution (see pseudocode in Algorithm 1).

Structured SVM algorithm aims to optimize the weight vector such that the dot product is an upper bound estimate of a loss function, i.e., unlike the perceptron it distinguishes between only partially and entirely incorrect solutions. A quadratic programming formulation capturing this requirement would demand exponentially many conditions for each of the training instances and its computation would be

Algorithm 1 The Structured Perceptron Algorithm

$\mathbf{w}_0 \leftarrow (0,\ldots,0)$
$\mathbf{w}_a \leftarrow (0,\ldots,0)$
$c \leftarrow 1$
for $i = 1 \ldots I$ **do**
  for $(\mathbf{x},\mathbf{y}^*) \in$ training set **do**
    $\hat{\mathbf{y}} \leftarrow \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}_\mathbf{x}} \mathbf{w}_0^T \Psi(\mathbf{x},\mathbf{y})$
    if $\mathbf{y}^* \neq \hat{y}$ **then**
      $\mathbf{w}_0 \leftarrow \mathbf{w}_0 + \Psi(\mathbf{x},\mathbf{y}^*) - \Psi(\mathbf{x},\hat{\mathbf{y}})$
      $\mathbf{w}_a \leftarrow \mathbf{w}_a + c\Psi(\mathbf{x},\mathbf{y}^*) - c\Psi(\mathbf{x},\hat{\mathbf{y}})$
    end if
    $c \leftarrow c + 1$
  end for
end for
return $\mathbf{w}_0 - \mathbf{w}_a / c$

intractable. For this reason, we use an iterative approximate algorithm which finds the most violated conditions in each iteration and add them as constraints to the quadratic programming problem.

Algorithm 2 The Cutting Plane Algorithm for Structured SVM

training data $\left((\mathbf{x}_1,\mathbf{y}_1^*),\ldots,(\mathbf{x}_n,\mathbf{y}_n^*)\right)$
$\mathbf{w} \leftarrow (0,\ldots,0)$
$\mathcal{W} \leftarrow \varnothing$
for $I = 1 \ldots I$ **do**
  $(\mathbf{w},\xi) \leftarrow \operatorname{argmin}_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$
    s.t. $\forall(\bar{\mathbf{y}}_1,\ldots\bar{\mathbf{y}}_n) \in \mathcal{W} : \frac{1}{n}\mathbf{w}^T \sum_1^n \left(\Psi(\mathbf{x}_i,\mathbf{y}_i^*) - \Psi(\mathbf{x}_i,\bar{\mathbf{y}}_i)\right) \geq \sum_i^n \Delta(\mathbf{y}_i^*,\bar{\mathbf{y}}_i) + \xi$
  for $i = 1 \ldots n$ **do**
    $\bar{\mathbf{y}}_i \leftarrow \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}_{\mathbf{x}_i}} \left(\Delta(\mathbf{y}^*) + \mathbf{w}^T\Psi(\mathbf{x}_i,\mathbf{y})\right)$
  end for
end for

The pseudocode is in Algorithm 2. The constraints are gradually added to the set $\mathcal{W}$. Line 7 of the algorithm contains loss-augmented inference, i.e., inference with respect to the current objective summed up with a loss function Δ. We use Hamming loss which allows us to do the loss-augmented inference efficiently – each edge not lying on the correct path is added a score of $\frac{1}{2}$.

Another approach we can take is to express the probability of the correct path using a log-linear model. Each path $\mathbf{y}$ in the hypothesis graph G is assigned a probability:

$$\mathsf{P}(\mathbf{y}|G) = \frac{\exp\left(\sum_{y\in\mathbf{y}} \mathbf{w}^T\varphi(y|G)\right)}{Z(G)}$$

where $Z(G)$ is the partition function, i.e. sum of the exponentiated scores of all path through the graph.

For computing the partition function, we can easily generalize the computation used for linear-chain CRFs. We define α_v as a sum of scores of all paths to a vector v. We can factorize the score for the edge from the rest of the summation for each

edge (u, v) going to this vertex. In this way we get a recursive formulation

$$\alpha(v) = \sum_{(u,v)\in E} \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \cdot \alpha(u).$$

In this way we can define $Z(G)$ as α of the target note and compute the value of the partition function efficiently using dynamic programming because the formula always utilizes only vertices that precede vertex in a topological ordering of the graph.

We define the optimization objective $\mathscr{L}$ as the average negative log likelihood of the training data $\mathscr{D}$:

$$\mathscr{L}(\mathscr{D}|\mathbf{w}) = -\sum_{(G,\mathbf{y}^*)\in\mathscr{D}} \log \mathsf{P}(\mathbf{y}^*|G).$$

The objective function $\mathscr{L}$ cannot be optimized in a closed form, however numerical methods using the gradient of the objective function could be used.

We compute the partial derivative of the loss function directly as follows

$$\frac{\partial \log \mathsf{P}(\mathbf{y}|G)}{\partial \mathbf{w}} = \sum_{y\in\mathbf{y}} \varphi(y|G) - \frac{1}{Z(G)} \cdot \frac{\partial Z(G)}{\partial \mathbf{w}}.$$

The partial derivative of the partition function can be derived directly from its recursive formulation using the function α.

$$\begin{aligned}
\frac{\partial\, \alpha(v)}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}} \sum_{(u,v)\in E} \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \cdot \alpha(u) \\
&= \sum_{(u,v)\in E} \frac{\partial}{\partial \mathbf{w}} \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \cdot \alpha(u) \\
&= \sum_{(u,v)\in E} \alpha(u) \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \varphi((u,v)|G) + \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \frac{\partial\, \alpha(u)}{\partial\, \mathbf{w}} \\
&= \sum_{(u,v)\in E} \exp\left\{\mathbf{w}^T \varphi((u,v)|G)\right\} \left[\alpha(u)\varphi((u,v)|G) + \frac{\partial\, \alpha(u)}{\partial\, \mathbf{w}}\right] \qquad (2.1)
\end{aligned}$$

This recursive formula gives us also an efficient way of computing the gradient via a dynamic programming algorithm.

We estimate the parameters using the stochastic gradient descent algorithm (Kushner and Clark, 1978).

Non-linearity in the scoring function can be introduced by adding a hidden layer and in fact scoring the edges by a Multi-layer Perceptron (MLP) which is left as a potential future work.

2.5.3 ILP Decoding

The decoding using the longest path algorithm presented in the previous sections credits the selected edges in a hypothesis graph with a cost and finds an overall maximum over all possible paths. However, does not employ any penalization for not including hypotheses which are very good, even though they do not lie on the extremal path. In this section we are going to introduce a model which adds some costs not only to those partial hypotheses selected by also to the rejected ones.

To satisfy these requirements, we redefine the decoding as a CRF. This CRF assigns a binary value from $\{0, 1\}$ to each vertex in the hypotheses graph G, indicating

whether the vertex label is part of the decoded solution. For that we treat the edges as undirected. Then we can assign a solution $\mathbf{y}$ of a hypotheses graph G energy function:

$$E(G,\mathbf{y}) = \sum_{v\in V} \mathbf{w}^{VT}\cdot\Phi_v(y_v,G) + \sum_{(u,v)\in E} \mathbf{w}^{ET}\Phi_{(u,v)}(y_u,y_v,G)$$

where Φ'_v are the vertices features and $\Phi'_{(u,v)}$ are the edges feature functions. Inference in such model is done by finding an assignment with the minimum energy value.

Whereas in the previous inference scheme we defined the feature functions only for the case the vertices were included into the decoded path, here we need to assign some feature function for situation when the vertices are not selected – a penalization for not using an edge. We define the energy function for edges as

$$\Phi'_{(u,v)}(y_u,y_v,G) = \begin{cases} \mathbf{w}_1^{ET}\Phi_{(u,v)}(G) & \text{if } y_u = y_v = 1 \\ \mathbf{w}_0^{ET}\Phi_{(u,v)}(G) & \text{otherwise} \end{cases},$$

and analogically the vertex feature functions. Note that setting the "otherwise" case to zero would mean the previous model.

The energy minimization itself does not guarantee that the boolean function assignment to the vertices would produce a feasible solution. This leads us to formulation the decoding as Constrained Conditional Model (CCM) (Roth and Yih, 2005). CCMs formulates the energy minimization as a integer program which allows hard constraints to be added to the model.

We can then formalize formulate the inference in the following way:

$$\max_{\mathbf{c}} \ \mathbf{c}^T\mathbf{x} \quad \text{, where}$$

$$\mathbf{x} = \underbrace{\{x_v\}_{v\in V}}_{\substack{\text{vertex}\\ \text{on the path}}} \quad \underbrace{\oplus\{x_u v\}_{(u,v)\in E}}_{\substack{\text{edge}\\ \text{on the path}}} \quad \underbrace{\oplus\{\bar{x}_v\}_{v\in V}}_{\substack{\text{vertex } \textit{not}\\ \text{on the path}}} \quad \underbrace{\oplus\{\bar{x}_u v\}_{(u,v)\in E}}_{\substack{\text{edge } \textit{not}\\ \text{on the path}}}$$

$$\mathbf{c} = \overbrace{\{\langle\phi^V(v,G),\mathbf{w}_1^V\rangle\}} \ \overbrace{\oplus\{\langle\phi^E(e,G),\mathbf{w}_1^E\rangle\}} \ \overbrace{\oplus\{\langle\phi^V(v,G),\mathbf{w}_0^V\rangle\}} \ \overbrace{\oplus\{\langle\phi^E(e,G),\mathbf{w}_0^E\rangle\}}$$

Subject to:

$$x_v + \bar{x}_v = 1 \quad \forall v \in V \tag{2.2}$$

$$x_{uv} + \bar{x}_{uv} = 1 \quad \forall (u,v) \in E \tag{2.3}$$

$$x_s, x_t = 1 \tag{2.4}$$

$$x_u - \sum_{\substack{v\in V\setminus\{t,u\}\\ (u,v)\in E}} x_{uv} = 1 \tag{2.5}$$

$$x_u - \sum_{\substack{v\in V\setminus\{s,u\}\\ (v,u)\in E}} x_{uv} = 1 \tag{2.6}$$

Fortunately, linear relaxation of the Integer Linear Programming (ILP) is guaranteed to have an integer solution, so the computation is efficient. This can be shown by induction according to the topological sort of the graph. The start node is guaranteed to have $x_s = 1$ explicitly. Now, assume all nodes which are before v in topological order of the graph – condition 2.6 for v and conditions 2.5 for the v's predecessors, does not allow any value of x_v except 0 and 1.

Whereas the perceptron learning procedure is directly applicable for this problem, the maximum margin learning requires defining the loss augmented inference.

Unlike the extremal path inference, we need to penalize both when an edge in incorrectly included to the path but also the situation when the edge is incorrectly excluded. For each edge, we update the feature vector to

$$\bar{\phi}(e) = \phi(e) \oplus (1 - \mathbf{1}[e \in \mathbf{y}^*], \mathbf{1}[e \in \mathbf{y}^*]),$$

and the weight vectors

$$\bar{\mathbf{w}}_1^e = \mathbf{w}_1^e \oplus \left(\frac{1}{2}, 0\right) \qquad \bar{\mathbf{w}}_0^e = \mathbf{w}_1^e \oplus \left(0, \frac{1}{2}\right).$$

We did not apply gradient learning for a maximum entropy formulation as in the case of the maximum path decoding. We did not solve how to efficiently compute the partition function and its gradient. As an alternative, negative sampling with Hinge loss could be used. This is left as a potential future work.

2.5.4 Results

The local decoding proved to be much stronger technique when we worked with the isolated words only. The results are tabulated in Table 2.1.

The original TextSpotter has accuracy for decoding 65% on our data and we consider this to be our baseline. In case of the isolated word decoding, the gradient based learning discriminating the correct path against all the others outperformed other optimization methods using only a cleverly selected contra-example as negative example. Non-linearity in features also appeared to be an important aspect in learning. If we classify the edges by a Random Forest classifier, we get over 70% accuracy which is itself comparable with the structured prediction approach. Combination of the non-linear classifier with other features gave us the best accuracy in maximum path decoding.

Using the ILP as inference algorithm proved to be more accurate that the maximum path decoding. We can expect that introducing gradient learning and non-linearity could even improve the results. However, the drawback of using ILP is its complexity. Unlike the maximum path which can be found in a linear time in number of vertices in a Directed Acyclic Graph (DAG), the complexity of linear programming is $O(n^{3.5})$ for the best know algorithms (Karmarkar, 1984) where n is number of variables, which is $2|E| + 2|V|$. This can be particularly problematic while computing n-best hypotheses.

On the other hand, the local decoding seems to fail in case in complete line decoding where it fails mostly in finding the correct word segmentation. In this case, the baseline cannot be easily estimated because the word splitting and decoding are done in different stages of the TextSpotter pipeline. There are 1.7 words per line which can give a very rough estimate of line accuracy as $65\%^{1.7} \approx 48\%$ assuming neighboring words correctness is independent on each other.

2.6 Decoding Based on Global Features

Using the local features that we explored in the previous section cannot capture longer distance features that would be obviously helpful for decoding. However, this way we cannot find a globally optimal solution.

	method	lines		words	
		accuracy	edit dist.	accuracy	edit dist.
Max Path	Structured perceptron	.489	1.037	.738	.463
	Structured perceptron + classifier	.500	1.047	.816	.377
	Structured SVM	—	—	.750	.439
	SGD	.500	.994	.778	.401
ILP	Structured perceptron	—	—	.829	.246
	Structured SVM	—	—	.810	.270

Table 2.1: Results of decoding using the local features. The lines are extracted from the ICDAR 2015 incidental dataset, the words form the ICDAR 2013 set.

2.6.1 Features for Global Decoding

Because the algorithm we use for exploring the space of string hypotheses is beam search, we can design the features telling not only how good the hypothesis as a whole can be, but also how the character we are about to add to the string fits into the previous part of the hypothesis. Slightly different set of features is then used in the rescoring phase.

The features can be categorized into few groups:

- *geometric consistency* – We expect the text on the same line to be geometrically consistent, i.e., the characters having similar width, height and to lie approximately the same line. Therefore we compute:
  - relative standard deviation of the character width, height and area;
  - relative standard deviation of space between the character with a space and without a space between them;
  - mean square error of fitting a line for center, top and bottom points of the character bounding boxes using a linear regression estimate.

  The features are used both in search and rescoring.
- *last character geometric fitting* – This should estimate how a character we are about to add to a hypothesis is geometrically consistent with the previous part of the hypothesis. We compute Gaussian divergence of its width, height and area with respect to the previous part of the hypothesis. We also compute the square error of fitting the bottom, top and center of new character's bounding box to lines fitted on the previous parts of the hypothesis.
- *language model* – We use a 7-gram character based model with standard EM smoothing estimated on texts from English Wikipedia operated on lowercased strings with numbers substituted by special tokens. We use word unigrams from one third of the data, bigrams from one third and trigrams from one third to also learn to locate the spaces in the texts. There is a special start- and end-of-string token. While doing the search, the end-of-string token is not used.
- *casing* – Another feature is whether is the text in standard casing, by which we mean entirely in lower-case or upper-case or capitalized. We also measure edit distance from the case-standardized version of the string and count of

suspicious substrings (`xX|x0|x0`, where `x`, and `X` mean lower- and upper-case letters, `0` mean a number). We also identified a set of garbage words that appear very often as substrings (`ii|iI|iI|lI|Il|il|li`) and introduced their presence as another feature.

- *TextSpotter features* – Some features from the original Textspotter decoding are reused. It is an overlap of the neighboring character extremal region thresholds and a score from a classifier telling whether there is a character on that position and its OCR score. The average value is used for both rescoring and search, values for the last character conjoined with including a or not-including a space.

- *crop coverage* – The preliminary experiments showed that the algorithm had tendencies to select a consistent and good-looking substring not caring that a lot characters were in fact skipped. Therefore we include four features which are: left and right margin size, how much of the rectangle envelope of the graph is covered by the selected letters and how much of the maximum string width is covered.

- *"technical" features* – Binary signs whether a space is added and whether this step terminates the sequence.

- *vocabulary features* – Because vocabularies are provided with some benchmarks, we would like to enforce the decoding to produce the dictionary words. On the other hand, we do not want to induce any hard limits because the "not care" words could have been recognized. We compute a proportion of words which are in the vocabulary. In case of search, we compute whether the last is a prefix of a vocabulary words.

- *spell checker features* – When a vocabulary is not provided we would also like to support hypotheses with existing words. We use a reimplementation of Aspell with U.S. English dictionary. The proportion of correct words and minimum edit distance from the first spell checker suggestion are used as features.

Computation of most of the features can be done in linear time in length of the hypothesis. If we cache the previously computed values, we can do additive changes in constant time.

2.6.2 Learning the Beam Search Scorer

During the beam search we score the hypotheses by a linear combination of the features we mentioned above. We use breadth-first search with the perceptron update rule which is a small modification of learning algorithm by Xu and Fern (2007).

The pseudocode of the algorithm is in Algorithm 3. We denote Φ the feature function for incomplete hypotheses, k is the size of the beam, the "BreadthExpand" function is one step of breadth-first search. We also experimented with passing a random beam to the next step instead of the k-best hoping it would learn a more robust function, but the results did not change. We use the beam size of 100 during during testing, however we used beam size of only 20 during training. Having a large beam increases the learning because the updates become less often, on the other hand when the beam size is too low we quickly overfit.

Algorithm 3 Learning Beam Search by the Perceptron Algorithm

for $i = 1 \ldots I$ **do**
  for $(\mathbf{x}, \mathbf{y}^*) \in$ training set **do**
    $B \leftarrow \{y_0^*\}$
    for $j = 1 \ldots |\mathbf{y}^*|$ **do**
      $B_{\text{next}} \leftarrow \cup_{\mathbf{h} \in B}$ BreadthExpand(h)
      $B \leftarrow k$ best $\mathbf{h} \in B_{\text{next}}$ by $\mathbf{w}^T \Phi(\mathbf{h})$
      $\mathbf{y}^*{}_i \leftarrow (y_0^*, \ldots y_i^*)$
      if $\mathbf{y}_\mathbf{i}^* \notin B$ **then**
        $\mathbf{w} \leftarrow \mathbf{w} + \left(\Phi(\mathbf{y}_i^*) - \frac{1}{|B|} \sum_{\mathbf{h} \in B} \Phi(\mathbf{h})\right)$
        $B \leftarrow B \cup \{\mathbf{y}_i^*\}$
      end if
    end for
  end for
end for

2.6.3 Hypotheses Rescoring

Because the way we learn the search scorer tries to ensure only that the correct hypothesis is present in the beam and the features describe mostly how the next character fits into the string, the first-best accuracy after the beam search decoding is obviously very low. Nevertheless in majority of cases, the correct word in the last beam of 100 hypotheses. This is the reason why we think rescoring of the final beam using an additional classifier could be helpful.

Comparing all hypotheses against each other would be computationally expensive. Therefore we use a binary classifier telling how likely the particular hypothesis to be correct by which evaluate each hypothesis only once. Features described in Section 2.6.1 are used, but we also include the maximum and minimum of each feature for the current n-best to encode also information about the other hypotheses.

We experimented with various classifiers from the WEKA toolkit, all having approximately the same performance. We report results with the SVM classifier with the radial basis kernel.

2.6.4 Results

The experiments with beam search decoding were done only for the complete line decoding. The experiments results are tabulated in Table 2.2.

As we stated before, the search itself performs relatively poorly, however the correct hypothesis is almost always present in the last beam. By the final rescoring we can get three times higher precision which is still not a satisfiable result.

The results also show that having a limited lexicon is a very helpful feature. Preferring words from a small lexicon of 150 words increases the performance by more than 10 percentage points.

If we do the same rough calculation as we did for the local decoding, we can estimate the accuracy for word decoding between 0.727 and 0.812 depending on the vocabulary size which is similar to the local decoding. From this we can hypothesize that for the single word decoding, introducing the global, longer distance features

method	acc. before rescoring	acc. after rescoring	n-best acc.	avg. edit dist.
small dictionary	.296	.736	.921	.682
medium dictionary	.229	.713	.906	.763
dictionary-free	.237	.624	.875	.977

Table 2.2: Results of decoding with beam search and final rescoring

does not help much. On the other hand for the complete line decoding, it seems it is crucial to achieve a convenient result.

2.7 Future Work

We see only minor improvements that can be done as a future work in the decoding task. One of them is using an efficient algorithm for getting n-best paths, e.g., by Kocan (2013). So far we have tried only Yen's algorithm (Yen, 1971) which becomes very slow with the growing size of n-best. In the gradient based learning, we could use additional hidden layers to bring some non-linearity to the model. Experiments with single words decoding showed that training a non-linear classifier for the edges independently and adding its score to the feature set can bring a significant improvement. Making the non-linearity an inherent part of the model can even better results. Both inference by ILP and formulating the model using the maximum entropy principle led to further improvements. Therefore combination of all of these might bring further improvements.

A non-linear scorer might be useful also for the beam search utilizing the global features. One way how it could be achieved is to use methods based on reinforcement learning, such as SEARN (Daumé III et al., 2009) or DAgger (Ross et al., 2011). We also hope such scorer would produce informative score that could improve also the final rescoring stage.

The final rescoring seems to perform relatively poorly. Although it seems to be a relatively easy task, its accuracy is still only about 80%. Despite not having any particular suggestions for this problem, we feel there is still a big room for improvement.

A technical, but still an important step is to fully integrate the decoder into the TextSpotter pipeline such that the improvements in decoding would became evident in the whole pipeline's performance.

3. Embeddings-based String Decoding

Unlike the previous chapter where the problem of forming words was decomposed to finding the best combination of recognized characters, in this chapter we would like to follow a different approach – try to avoid explicit decomposition of the of the words to letters and embed the words to the same vector space as the images.

This chapter starts with a brief overview of related work. Next section, describes our preliminary experiments with synthetic data generation. Section **??** contains description of two methods for text recognition we propose. One uses a dictionary, the second one is lexicon-free.

Everything mentioned in this chapter is a work in progress or a future work, therefore no results of any experiments are presented.

3.1 Related Work

The so called embeddings – a continuous vector representation of various objects – became an intensively studied research topic.

While talking about word embeddings people usually mean word representation that was inferred during training of neural language model (Bengio et al., 2006; Collobert and Weston, 2008; Mikolov, 2012). Mikolov et al. (2013) showed that there is actually no need to accurately estimate the probability of the following words and a very rough approximation is enough to infer a very good continuous space representation. It also appeared that these embeddings can also be produced by a word co-occurrence matrix factorization (Levy and Goldberg, 2014).

Word embeddings proved to be useful in almost all areas of NLP. The most startling property of the word embeddings, i.e., that it is able to efficiently capture not only morphological and syntactic properties but also semantic and pragmatic properties. Paragraph embeddings derived from word2vec (Mikolov et al., 2013) produces a representation that outperforms all previous method in the task of sentiment analysis (Le and Mikolov, 2014).

The objects projected to a continuous space do not necessarily be words or sentences, but these can be also smaller units as characters or character n-grams. We can easily assume that semantics of the words does not influence the way they are displayed in scenes. The classical word embeddings also usually neglect that the words themselves consist of smaller units. For tasks where the inner word structure plays important role, the character embeddings are used, like text segmentation (Chrupala, 2013), morphological tagging (Santos and Zadrozny, 2014).

This boom of methods mentally related to deep learning is not limited to NLP. In the machine vision area, the ImageNet (Krizhevsky et al., 2012), network holds a similar status as word2vec in NLP. The last layer before the final soft-max layer is very often used as an image descriptor.

Continuous representations have also been used for projecting content of different modalities. Vinyals et al. (2014) achieved a fascinating result with a recursive NN for generating description of images from an image representation. Habibian et al. (2014) developed joined embeddings of events in videos and their tags which was used for automatic tagging of events in videos

In the context of STR, there are only two works which try to embed the image and labels into the same vector space and none of them uses NNs. Rodriguez and Perronnin (2013) represents words as a histograms of characters in the whole word, in the halves of the word and in the quarters of the word (i.e, concatenation of 7 frequency vectors). For the image representation they use a Gaussian mixture of what they call patches. These representations are linearly projected to the same space with the projection matrices estimated by structured SVM. Almazan et al. (2014) uses similar word representation, SIFT descriptors for the image and the projection is estimated using canonical correlation analysis.

We can assume that image representation produced by CNN of Jaderberg et al. (2014a) can have very similar properties and also produces some kind of general image representation. We can also expect that at the upcoming events, more such papers will appear.

3.2 Synthetic Training Data

Because the NN-based methods demand a lot of training data which is very difficult to obtain, we need to look for other sources of necessary amount of annotated training data for supervised learning. Using synthetic data is a common practice in OCR and until recently it has not been used in STR.

Our approach is very much inspired by Jaderberg et al. (2014a), however there are some minor differences. These are caused mainly because whereas Jaderberg et al. (2014a) is primarily interested in recognition of words from a limited vocabulary, we are interested in training a general vector representation of cropped scene text.

To generate synthetic data, we first decide what text will be plotted in the image. We sample n-grams sampled from the Common Crawl corpus. We select randomly one font of 1,700 Google fonts[1]. At first we render the text in black and white, and generated randomly its rotation and perspective distortion. From this, we know how big the final image will be.

In the second stage we randomly choose an image from the Street View Dataset (Wang and Belongie, 2010) and do the k-means clustering on the image pixels. In this way we get 16 representative colors from the image. Later we chose a random crop from the image which will be the actual background of the text. We sample the text color from for representative colors which are most dissimilar to the selected background.

A simulation of a flash can be randomly added to the image as a white circle with concentrically decreasing transparency. The second type distortion we add to the images is placing randomly cropped rectangles from a Street View photo over the text. Finally, Gaussian noise of random intensity is added.

All parameters of distribution and other constants are estimated empirically based on visual feedback.

The current experiments show that it is hard to find a compromise between high variability of the dataset and having the images readable and realistic. Therefore we plan to annotate dozens of generated examples and train a CNN telling whether a generated image looks realistic or not. A similar approach can be taken for deciding

[1] http://www.google.com/fonts

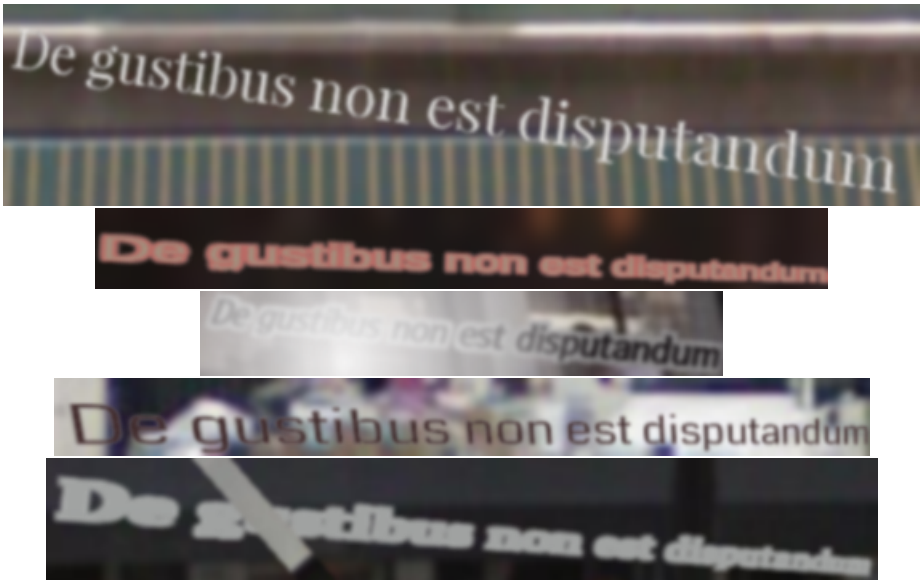

Figure 3.1: Examples of synthetically generated scene text.

whether the n-grams sampled from the corpus are or are not a good example of a scene text. Any annotation we could get for the data will be very noisy, but hopefully some already existing machine learning method that can deal with such noisy annotation (Tibshirani and Manning, 2014).

3.3 Future Work

There are two ideas based on CNN we would like to explore in detail: One is to use visual similarity of images for choosing words from a lexicon; the second one is to use a search algorithm comparing string to part of the image for lexicon-free recognition.

In both cases, we would like to use CNN which can deal with images of arbitrary size. We hope that the convolutional filter can learn to detect events as presence of particular characters or character n-grams which will provide strong clues for decisions about the string. The classical max-pooling which is used in CNNs only captures information about existence of something, but in case of strings, the relative position of the maxima might be important. This is why we think about adding also the argmax value in the pooling as was used by Fonseca and Rosa (2013) in a CNN for semantic role labeling.

The first idea is to use a NN for comparison of an image with synthetic images having the same string. During the learning phase we will generate a few image examples for each word from the vocabulary and then learn a classifier telling whether the image we want to classify contains the same text as the others with shared weights for all of the images. It this way, we hope we can get a universal representation of the images that can be reused with arbitrary vocabularies. Another advantage is that in the test phase, the vocabulary vectors can be precomputed and the vector representation of the tested image will also be computed also only once, regardless the size of the vocabulary.

For the second method, we would like to use a search algorithm as DAgger (Ross et al., 2011) to search through the space of possible string. Similarly as in case of bottom-up decoding, we think it would be useful to work with complete lines of text because longer language context can bring more information from language modeling. The scoring function of DAgger will have three inputs: a vector representation of the image, a vector representation of the evaluated string from CNN about which we hope it will learn the orthograhical properties of the string and vector representation from a language model.

Note that if we build the hypothesis left to right, the convolutional filters can be update only be processing the added part of the image and there the feature extraction will be computed in a linear time.

Conclusion & Future Work

So far, we focused on getting familiar with STR research via doing experiments with the string decoding, which is a task very much related to what is done in speech recognition or machine translation. At the beginning, we thought that bringing language modeling techniques as well as state-of-the-art of structured prediction methods will help to improve the STR performance. Moreover, we thought it could be a good foundation for solving the task in which order the words in a scene should be read in and which words belong together. We submitted the results of our decoding experiments to the ICDAR 2015 Robust Reading competition, however we expect this will be outperformed by more recent methods based on deep learning.

The recent publication of the very successful methods based on deep learning led to re-evaluation of our original plans. As described in Chapter 3, we would like to develop a neural-network-based method for cropped text recognition which will be still lexicon-free but with a significant role of language model that will the model capability to deal with noise and obstacles in the image.

In the text of the proposal, we have not yet discussed in detail the more linguistically motivated task – connecting the words in the image to meaningful groups and read them in the order people would do. It is a task that has not been approached before because researchers probably assume it could be done by a simple heuristic and do not consider this task to be important. Further processing of the scene text as machine translation or information retrieval would strongly benefit from having input in such a format.

There is currently no dataset annotated with this information. Approximately we plan to start to enrich annotation of the ICDAR 2015 dataset of both focused and incidental text.

A suitable metric for the task also needs to be developed. We plan to conduct an experiment with several variants of metrics we have designed to find out which one correlates the most with the human judgment. The people in the experiment will see transcriptions of the image text with artificially added noise and will be asked to evaluate how good they think the transcription is. From this experiment, we will find out not only how the metrics correlate with human judgment but also to what errors are people more sensitive.

Bibliography

J. Almazan, A. Gordo, A. Fornes, and E. Valveny. Word spotting and recognition with embedded attributes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(12):2552–2566, Dec 2014. ISSN 0162-8828.

Morgan Ames and Mor Naaman. Why we tag: Motivations for annotation in mobile and online media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 971–980, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-593-9.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In DawnE. Holmes and LakhmiC. Jain, editors, *Innovations in Machine Learning*, volume 194 of *Studies in Fuzziness and Soft Computing*, pages 137–186. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-30609-2.

Tim Berners-Lee and Robert Cailliau. WorldWideWeb: Proposal for a HyperText Project. Technical report, CERN, 1990. URL http://www.w3.org/Proposal.

Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. Photoocr: Reading text in uncontrolled conditions. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 785–792. IEEE, 2013.

Axel Bruns. Produsage. In *Proceedings of the 6th ACM SIGCHI Conference on Creativity &Amp; Cognition*, C&C '07, pages 99–106, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-712-4.

De Campo, Babu, and Varma. Character Recognition in Natural Images. 2009.

John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

Huizhong Chen, Sam S Tsai, Georg Schroth, David M Chen, Radek Grzeszczuk, and Bernd Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 2609–2612. IEEE, 2011.

Grzegorz Chrupala. Text segmentation with character-level text embeddings. *CoRR*, abs/1309.4628, 2013.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics, 2002.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4.

Robert T. Craig. Communication Theory as a Field. *Communication Theory*, 9: 2: 119–161, 1999.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

Hal Daumé III, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine learning*, 75(3):297–325, 2009.

T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In *Proceedings of the International Conference on Computer Vision Theory and Applications, Lisbon, Portugal*, 2009.

Florian Deckert, Benjamin Seidler, Markus Ebbecke, and Michael Gillmann. Table content understanding in smartfix. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 488–492. IEEE, 2011.

Vikram Dendi. Translator app now available for Windows Phone 8, 2012. URL http://blogs.msdn.com/b/translation/archive/2012/11/26/translator-app-now-available-for-windows-phone-8.aspx.

Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970. IEEE, 2010.

Facebook, Inc. Focusing on efficiency: A whitepaper from Facebook, Ericsson and Qualcomm. Technical report, Sep 2013. URL https://fbcdn-dragon-a.akamaihd.net/hphotos-ak-prn1/851575_520797877991079_393255490_n.pdf.

Jacqueline Field. *Improving Text Recognition in Images of Natural Scenes*. PhD thesis, University Massachusetts Amherst, February 2014.

E.R. Fonseca and J.L.G. Rosa. A two-step convolutional neural network approach for semantic role labeling. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–7, Aug 2013.

Vincent Gaudissart, Silvio Ferreira, Céline Thillou, and Bernard Gosselin. SYPOLE: mobile reading assistant for blind people. In *9th Conference Speech and Computer*, 2004.

R. Geuss. *The Idea of a Critical Theory: Habermas and the Frankfurt School*. Modern European Philosophy. Cambridge University Press, 1981. ISBN 9780521284226.

V.K Govindan and A.P Shivaprasad. Character recognition – A review. *Pattern Recognition*, 23(7):671 – 683, 1990. ISSN 0031-3203.

Amirhossein Habibian, Thomas Mensink, and Cees G.M. Snoek. VideoStory: A new multimedia embedding for few-example recognition and translation of events. In *Proceedings of the ACM International Conference on Multimedia*, MM '14, pages 17–26, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3063-3.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

Ismail Haritaoglu. Scene text extraction and translation for handheld devices. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–408. IEEE, 2001.

Jennifer Howard. Google begins to scale back its scanning of books from university libraries. ISSN 0009-5982. URL https://chronicle.com/article/Google-Begins-to-Scale-Back/131109/.

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

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *Computer Vision–ECCV 2014*, pages 512–528. Springer, 2014b.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

Jehyun Jung, SeongHun Lee, Min Su Cho, and Jin Hyung Kim. Touch TT: Scene text extractor using touchscreen interface. *ETRI Journal*, 33(1):78–88, 2011.

Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, Lluis Pere de las Heras, et al. ICDAR 2013 robust reading competition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1484–1493. IEEE, 2013.

N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, pages 302–311, New York, NY, USA, 1984. ACM. ISBN 0-89791-133-4.

Fatih Kocan. A nonenumerative algorithm to find the k longest (shortest) paths in a DAG. *CoRR*, abs/1301.0181, 2013.

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

Harold Joseph Kushner and Dean S. Clark. *Stochastic approximation methods for constrained and unconstrained systems*. Applied mathematical sciences. Springer, New York, 1978. Includes index.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Tony Jebara and Eric P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196. JMLR Workshop and Conference Proceedings, 2014.

SeongHun Lee, Min Su Cho, Kyomin Jung, and Jin Hyung Kim. Scene text extraction with edge constraint and text collinearity. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3983–3986. IEEE, 2010.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc., 2014.

Pierre Lévy and Robert Bonomo. *Collective intelligence: Mankind's emerging world in cyberspace*. Perseus Publishing, 1999.

Jindřich Libovický, Lukáš Neumann, Pavel Pecina, and Jiří Matas. A machine learning approach to hypothesis decoding in scene text recognition. In *Computer Vision-ACCV 2014 Workshops*, pages 169–180. Springer, 2014.

Eden Litt and Eszter Hargittai. Smile, snap, and share? A nuanced approach to privacy and online photo-sharing. *Poetics*, 42:1–21, 2014.

Xiaoqing Liu and Jagath Samarabandu. Multiscale edge-based text extraction from complex images. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1721–1724. IEEE, 2006.

Simon M Lucas. ICDAR 2005 text locating competition results. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 80–84. IEEE, 2005.

Simon M Lucas, Alex Panaretos, Luis Sosa, Anthony Tang, Shirley Wong, Robert Young, Kazuki Ashida, Hiroki Nagai, Masayuki Okamoto, Hiroaki Yamamoto, et al. ICDAR 2003 robust reading competitions: entries, results, and future directions. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3):105–122, 2005.

Bo Luo, Xiaogang Wang, and Xiaoou Tang. World Wide Web based image search engine using text and image content features. In *Electronic Imaging 2003*, pages 123–130. International Society for Optics and Photonics, 2003.

Céline Mancas-Thillou and Bernard Gosselin. Natural scene text understanding. In Goro Obinata and Ashish Dutta, editors, *Vision Systems: Segmentation and Pattern Recognition*. I-Tech Education and Publishing, Vienna, 2007.

Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22 (10):761–767, 2004.

M. McLuhan. *The Gutenberg Galaxy: The Making of Typographic Man*. Canadian university paperbooks. University of Toronto Press, 1962. ISBN 9780802060419.

Tomáš Mikolov. *Statistical Language Models Based on Neural Networks*. PhD thesis, FIT VUT, Brno, 2012.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Jonathan Fabrizio, and Beatriz Marcotegui. Snoopertext: A multiresolution system for text detection in complex visual scenes. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 3861–3864. IEEE, 2010.

A. Mishra, K. Alahari, and C. V. Jawahar. Scene text recognition using higher order language priors. In *BMVC*, 2012a.

Anand Mishra, Karteek Alahari, and CV Jawahar. Top-down and bottom-up cues for scene text recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2687–2694. IEEE, 2012b.

Anand Mishra, Karteek Alahari, and CV Jawahar. Image retrieval using textual cues. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3040–3047. IEEE, 2013.

Shunji Mori, Ching Y Suen, and Kazuhiko Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80(7):1029–1058, 1992.

Robert Nagy, Anders Dicker, and Klaus Meyer-Wegener. NEOCR: A configurable dataset for natural image text recognition. In *Camera-Based Document Analysis and Recognition*, pages 150–163. Springer, 2012.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4, 2011.

Lukáš Neumann and Jiří Matas. Real-time scene text localization and recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3538–3545, California, US, 2012. IEEE. ISBN 1063-6919.

Lukáš Neumann and Jiří Matas. On combining multiple segmentations in scene text recognition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 523–527. IEEE, 2013.

Tatiana Novikova, Olga Barinova, Pushmeet Kohli, and Victor Lempitsky. Large-lexicon attribute-consistent text recognition in natural images. In *Computer Vision–ECCV 2012*, pages 752–765. Springer, 2012.

A Oeldorf-Hirsch and SS Sundar. Online photo sharing as mediated communication. In *Proc. of the Annual Conference of the International Communication Association, ICA, Washington*, 2010.

Wen-wu Ou, Jun-min Zhu, and Chang-ping Liu. Text location in natural scene. *Journal of Chinese Information Processing*, 5:006, 2004.

Leon Palm and Jiayong Zhang. Google Goggles gets faster, smarter and solves Sudoku, 2011. URL http://googleblog.blogspot.cz/2011/01/google-goggles-gets-faster-smarter-and.html.

Wumo Pan, TD Bui, and CY Suen. Text detection from natural scene images using topographic maps and sparse representations. In *IEEE ICIP*, 2009.

Yi-Feng Pan, Cheng-Lin Liu, and Xinwen Hou. Fast scene text localization by learning-based filtering and verification. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 2269–2272. IEEE, 2010.

Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu. A hybrid approach to detect and localize texts in natural scene images. *Image Processing, IEEE Transactions on*, 20 (3):800–813, 2011.

Sirous Panahi, Jason Watson, and Helen Partridge. Social media and tacit knowledge sharing: developing a conceptual model. *World academy of science, engineering and technology*, (64):1095–1102, 2012.

Sarah Perez. Google now using recaptcha to decode street view addresses, March 2012. URL http://techcrunch.com/2012/03/29/google-now-using-recaptcha-to-decode-street-view-addresses.

Nicole A Poltash. Snapchat and sexting: A snapshot of bearing your bare essentials. *Rich. JL & Tech.*, 19:14–14, 2013.

Ingmar Posner, Peter Corke, and Paul M Newman. Using text-spotting to query the world. In *IROS*, volume 10, pages 3181–3186, 2010.

N. Postman. *Amusing Ourselves to Death: Public Discourse in the Age of Show Business*. Penguin Books, 2006. ISBN 9780143036531.

Sheikh Faisal Rashid, Faisal Shafait, and Thomas M Breuel. Visual recognition of permuted words. In *IS&T/SPIE Electronic Imaging*, pages 752715–752715. International Society for Optics and Photonics, 2010.

Keith Rayner, Sarah J White, Rebecca L Johnson, and Simon P Liversedge. Raeding wrods with jubmled lettres there is a cost. *Psychological science*, 17(3):192–193, 2006.

S.V. Rice, F.R. Jenkins, T.A. Nartker, and Las Vegas) Information Science Research Institute (University of Nevada. *The fifth annual test of OCR accuracy*. Information Science Research Institute, 1996.

Jose Rodriguez and Florent Perronnin. Label embedding for text recognition. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2013.

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

Dan Roth and Wen-tau Yih. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd international conference on Machine learning*, pages 736–743. ACM, 2005.

Sangheeta Roy, Partha Pratim Roy, Palaiahnakote Shivakumara, Georgios Louloudis, Chew Lim Tan, and Umapada Pal. HMM-based multi oriented text recognition in natural scene image. In *Pattern Recognition (ACPR), 2013 2nd IAPR Asian Conference on*, pages 288–292. IEEE, 2013.

U. Roy, A. Mishra, K. Alahari, and C. V. Jawahar. Scene text recognition and retrieval for large lexicons. In *Proceedings of the Asian Conference on Computer Vision*. Springer, 2014.

Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. LabelMe: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

Cicero D. Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In Tony Jebara and Eric P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826. JMLR Workshop and Conference Proceedings, 2014.

Herbert F Schantz. *History of OCR, optical character recognition*. Recognition Technologies Users Association, 1982.

Daniel Schuster, Klemens Muthmann, Daniel Esser, Alexander Schill, Michael Berger, Christoph Weidling, Kamil Aliyev, and Andreas Hofmeier. Intellix–end-user trained information extraction for document archiving. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 101–105. IEEE, 2013.

Asif Shahab, Faisal Shafait, and Andreas Dengel. ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1491–1496. IEEE, 2011.

Claude Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

Cunzhao Shi, Chunheng Wang, Baihua Xiao, Yang Zhang, Song Gao, and Zhong Zhang. Scene text recognition using part-based tree-structured character detection. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2961–2968. IEEE, 2013.

Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

Arnold WM Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(12):1349–1380, 2000.

Apurva Srivastav and Jayant Kumar. Text detection in scene images using stroke width and nearest-neighbor constraints. In *TENCON 2008-2008 IEEE Region 10 Conference*, pages 1–5. IEEE, 2008.

Bolan Su and Shijian Lu. Accurate scene text recognition based on recurrent neural network. In *Proceedings of the Asian Conference on Computer Vision*. Springer, 2014.

Céline Thillou, Silvio Ferreira, and Bernard Gosselin. An embedded application for degraded text recognition. *EURASIP Journal on applied signal processing*, 2005: 2127–2135, 2005.

Julie Tibshirani and D. Christopher Manning. Robust logistic regression using shift parameters. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 124–129. Association for Computational Linguistics, 2014.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and Tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.

Luis von Ahn. Human computation. In *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE*, pages 418–419, July 2009.

Luis von Ahn and Will Cathcart. Teaching computers to read: Google acquires recaptcha, 2009. URL http://googleblog.blogspot.cz/2009/09/teaching-computers-to-read-google.html.

Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

Jakob Voss. Tagging, folksonomy & co – renaissance of manual indexing? *CoRR*, abs/cs/0701072, 2007.

Hao Wang and Jari Kangas. Character-like region verification for extracting text in scene images. In *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*, pages 957–962. IEEE, 2001.

Kai Wang and Serge Belongie. Word spotting in the wild. In *Proceedings of the 11th European conference on Computer vision: Part I*, pages 591–604. Springer, 2010.

Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1457–1464. IEEE, 2011.

Mark Warschauer and Douglas Grimes. Audience, authorship, and artifact: The emergent semiotics of Web 2.0. *Annual Review of Applied Linguistics*, 27:1–23, 2007.

J.J. Weinman, Z. Butler, D. Knoll, and J. Feild. Toward integrated scene text reading. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(2):375–387, Feb 2014. ISSN 0162-8828.

Yuehua Xu and Alan Fern. On learning linear ranking functions for beam search. In *Proceedings of the 24th international conference on Machine learning*, pages 1047–1054. ACM, 2007.

Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1083–1090. IEEE, 2012.

Qixiang Ye, Jianbin Jiao, Jun Huang, and Hua Yu. Text detection and restoration in natural scene images. *Journal of Visual Communication and Image Representation*, 18(6):504–513, 2007.

Jin Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.

Chucai Yi and YingLi Tian. Text string detection from natural scenes by structure-based partition and grouping. *Image Processing, IEEE Transactions on*, 20(9): 2594–2605, 2011.

Xu-Cheng Yin, Xuwang Yin, and Kaizhu Huang. Robust text detection in natural scene images. *CoRR*, abs/1301.2628, 2013.

Honggang Zhang, Kaili Zhao, Yi-Zhe Song, and Jun Guo. Text extraction from natural scene image: A survey. *Neurocomputing*, 122:310–323, 2013.

Gang Zhou, Yuehu Liu, Quan Meng, and Yuanlin Zhang. Detecting multilingual text in natural scene. In *Access Spaces (ISAS), 2011 1st International Symposium on*, pages 116–120. IEEE, 2011.

Øivind Due Trier, Anil K. Jain, and Torfinn Taxt. Feature extraction methods for character recognition – A survey. *Pattern Recognition*, 29(4):641 – 662, 1996. ISSN 0031-3203.