A Comprehensive Neural-Based Approach for Text Recognition in Videos using Natural Language Processing

Khaoula Elagouni, Christophe Garcia, Pascale Sébillot

To cite this version:

Khaoula Elagouni, Christophe Garcia, Pascale Sébillot. A Comprehensive Neural-Based Approach for Text Recognition in Videos using Natural Language Processing. ACM International Conference on Multimedia Retrieval, ICMR, Apr 2011, Trento, Italy. 8 p., 2 columns. hal-00645219

HAL Id: hal-00645219
https://hal.archives-ouvertes.fr/hal-00645219

Submitted on 27 Nov 2011

A Comprehensive Neural-Based Approach for Text Recognition in Videos using Natural Language Processing

Khaoula Elagouni
Orange Labs R&D
4 rue du Clos Courtel
F-35512 Cesson-Sévigné
Cedex, France
khaoula.elagouni@orange-ftgroup.com

Christophe Garcia
LIRIS, Insa de Lyon
Bât. Jules Verne
17 avenue Jean Capelle
F-69621 Villeurbanne Cedex,
France
christophe.garcia@liris.cnrs.fr

Pascale Sébillot
IRISA, Insa de Rennes
Campus de Beaulieu
F-35042 Rennes Cedex,
France
pascale.sebillot@irisa.fr

ABSTRACT

This work aims at helping multimedia content understanding by deriving benefit from textual clues embedded in digital videos. For this, we developed a complete video Optical Character Recognition system (OCR), specifically adapted to detect and recognize embedded texts in videos. Based on a neural approach, this new method outperforms related work, especially in terms of robustness to style and size variabilities, to background complexity and to low resolution of the image. A language model that drives several steps of the video OCR is also introduced in order to remove ambiguities due to a local letter by letter recognition and to reduce segmentation errors. This approach has been evaluated on a database of French TV news videos and achieves an outstanding character recognition rate of 95%, corresponding to 78% of words correctly recognized, which enables its incorporation into an automatic video indexing and retrieval system.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*indexing methods, linguistic processing*; H.2.4 [**Database management**]: Systems—*multimedia databases*; I.5.1 [**Pattern Recognition**]: Models—*neural nets*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*language models*

Keywords

Video Indexing, Video OCR, Text Extraction from Videos, Convolutional Neural Networks, Language Model.

1. INTRODUCTION

The volumes of available audio-visual documents continue to grow especially with the advent of photo and video sharing, delinearized television programs and video on-demand professional systems. In this context, multimedia indexing and retrieval becomes an issue of great importance.

During the last years, a lot of work has been dedicated to the problem of indexing videos by automatically analyzing their contents. Digital video contents are described by means of some extracted objects considered as semantically important [16], or of sets of key frames [4], by classifying shots and events [19], or taking into account speech transcription [5]. Other approaches [15] make use of text embedded in videos as another way to access the semantics of these multimedia documents. So does our work which aims at providing an accurate method to extract textual clues artificially embedded in videos. These texts—which correspond to report titles, city or guest names, scores, etc.—are indeed strong semantic clues, and thus important elements for video indexing and retrieval. However, they can be variable in size, color and form (fonts) and appear on textured backgrounds, in low-resolution images, especially in the case of videos. These difficulties make the recognition task more challenging. In this paper, we present a comprehensive video Optical Character Recognition (OCR) system, specifically adapted to video data, that detects, extracts and recognizes embedded texts. In addition to the robustness of our recognition method relying on a neural classification approach, our second contribution lies in the introduction of a supervision scheme based on a language model.

The rest of this paper is organized as follows. After a state of the art (Section 2) dedicated to the different steps of a video OCR system, our own OCR is detailed in Section 3. The results of its evaluation on a database of TV news videos are presented and discussed in Section 4. Finally, Section 5 provides some conclusions and highlights future work.

2. RELATED WORK

Until the nineties, most researches focused on OCR systems operating on scanned documents. During recent years however, considerable progress, initiated by Lienhart et al. [15] in 1996, has been made in the recognition of text embedded in digital videos. In Lienhart's work, considering some properties of embedded text, frames are segmented in order to identify candidate text regions. An OCR is then performed to discard non-text regions and recognize characters. The character recognizer is however not completely adapted to the characteristics of embedded text; thus results are unsatisfactory. Next methods focused on preprocessings useful to increase extraction performances of commercial document

ICMR '11, April 17-20, Trento, Italy

OCR. Chen et al. [6] introduced a step of binarization of the text image before the OCR, and garbage characters ('.', ':', '!', etc.) were removed by means of a language model. The weakest step remains the binarization, especially in the case of a complex background, that generally leads to a loss of information often crucial for the recognition. Commercial OCRs are indeed usually developed for printed text, much less variable in style, color and size than texts in videos. Hua et al. [11] and Yi et al. [23] were interested in solving problems related to complex backgrounds by using multiple frame integration. The main idea consists in taking advantage of the temporal redundancy of a text in numerous frames.

Work has also been dedicated to the development of single character recognizers in images and videos. Two main approaches can be distinguished: pattern matching methods and machine learning methods. In the first category, characters are usually identified by a set of features. First, a database of feature models is generated. Then, for each image corresponding to a character, features are extracted and matched to the database in order to recognize the character class. Edges and contours are considered as features in [12], whereas side profiles are taken into account in [7]. In [24], after extracting character features, a Bayesian framework is used to combine various information and recognize characters. Recently, inspired by speech recognition, Som et al. [20] defined an OCR system that uses a Hidden Markov Model to identify characters as a sequence of states. However, as in any pattern recognition problem, the major issue of these methods is to determine the discriminant features to represent the characters. Performances highly depend on this choice. In the second category, methods learn automatically to classify characters either directly from their images or after extracting features. Dorai et al. [9] rely on a Support Vector Machine (SVM) classifier to learn how to categorize characters and obtain satisfactory results. Saïdane and Garcia [17] propose an automatic method for scene character recognition based on a convolutional neural classification approach, which outperforms existing methods.

In this paper, we propose a novel and robust scheme for text recognition in videos, with a convolutional neural approach for character recognition. This framework is here adapted to our video data, taking benefit from text temporal redundancy. We show that this approach produces outstanding recognition rates, and outperforms SVM methods for single character recognition (cf. Section 4). Moreover, in order to overcome the limits of this local character by character recognition, a language model is introduced to take into account dependencies between successive characters.

3. TEXT RECOGNITION SCHEME

In this section, we depict the outlines of our recognition scheme. As shown in fig. 1, our approach involves five processing steps. The following subsections describe each of them and their interactions within the recognition scheme.

3.1 Text Detection and Tracking

The first step of our recognition scheme consists in locating and extracting texts embedded in videos. Delakis and Garcia [8] provided a solution for horizontal text detection in images. This robust method relies on a convolutional neural network and outperforms current state-of-the-art approaches [8]. We choose to adapt it to our video data case.

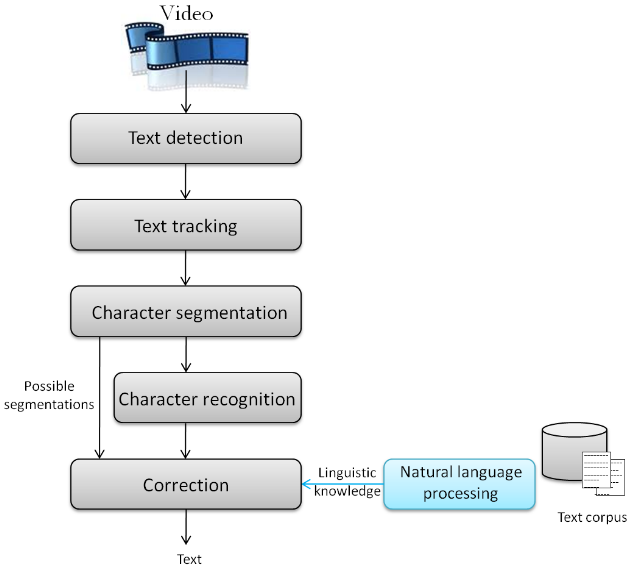

Figure 1: The proposed video OCR scheme.

As shown in fig. 2, we apply a detection step every two seconds[1] in order to locate new texts that are going to appear and those previously displayed that are going to disappear. Since static texts in videos are considered, a tracking process that determines the times of the start and the end of each localized text (cf. fig. 2) is introduced. This task is ensured by a similarity measure between the image of the detected text and images appearing in the same location during the 2 second sequence. The intensity correlation is used as an accurate indicator to evaluate the visual similarity. Correlation images are then analyzed in order to decide if correlated images correspond to the same text or not.

3.2 Enhanced Character Segmentation Based on the Shortest Path Algorithm

Before recognizing the texts, a preliminary segmentation step is necessary in order to obtain one image per individual character. This segmentation step is crucial for the character recognition: any error directly reduces the recognition accuracy of the OCR system (see [2] for a state of the art of text segmentation methods).

In our work, we perform a segmentation that permits to separate characters depending on their morphologies and deriving benefit from the temporal redundancy of texts in videos.

3.2.1 *Statistical Intensity Analysis*

In order to correctly segment the characters, we need first to identify the background and to distinguish it from the text. Pixels of images corresponding to texts are of two classes: "text" and "background". We admit that intensities of text regions and background are both governed by gaussian distributions. Using the Expectation-Maximization algorithm (EM), we formalize the problem as maximizing the likelihood between a set of observations—namely the image

[1] We made the assumption that in order to be legible, texts embedded in videos must appear at least for two seconds.

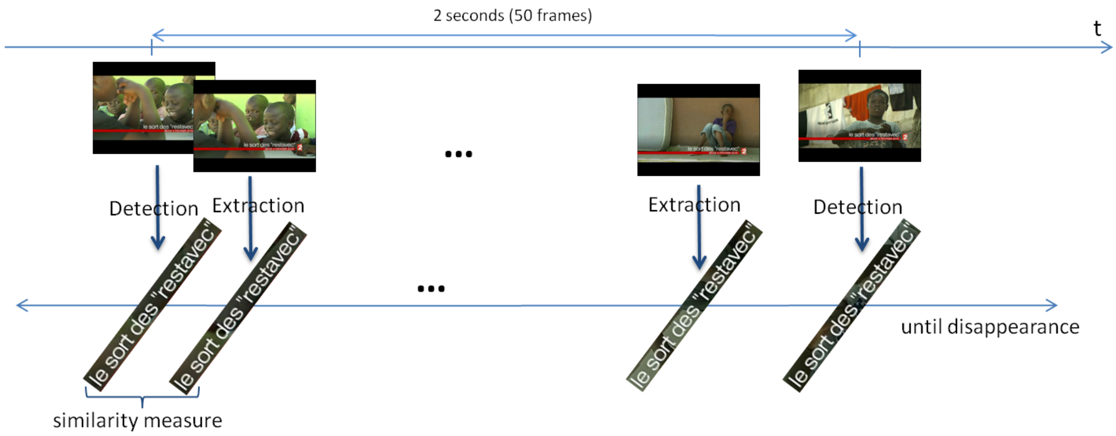

Figure 2: Text detection and tracking process in videos.

histogram—and a gaussian mixture model. Parameters of obtained distributions are then used to generate a fuzzy map where each pixel is represented by its membership degree to the class "text". In addition to the intensity analysis, we introduce the multiple frame integration. Since texts generally keep the same visual attributes from their appearance to their disappearance, background can be identified by its possible temporal variability. Each pixel is treated separately to evaluate its temporal standard deviation. Using these values, another fuzzy map indicating the membership degrees to the class "background" is generated.

Since intensity distributions and temporal variation represent two independent sources of information, we combine the two fuzzy maps described above, in order to obtain a more accurate membership map. For this, a fusion system, with an adaptive behavior depending on values to combine, is required. According to the definitions in [22], the chosen operator should be conjunctive (with a severe behavior) if both values are low, disjunctive (with an indulgent behavior) if both values are high and depends only on intensity distribution analysis if the temporal variation is low. The operator expressed by eq. 1 satisfies these conditions:

$$f\,(x,y) = \begin{cases} x & if\ y \leq th \\ \sigma(x,y) = \frac{g(x,y)}{g(x,y)+g(1-x,1-y)} & otherwise \end{cases} \qquad (1)$$

where x refers to the intensity analysis result, y refers to the temporal variation result and th is a threshold determined empirically. $\sigma(x,y)$ is the associative symmetric sum, and $g(x,y)$ is a positive increasing function. Fig. 3 illustrates an example of generated map. These fuzzy maps are used for character segmentation. Gray scale images, which contain further important information for the character recognition task, are however considered for the rest of the OCR steps.

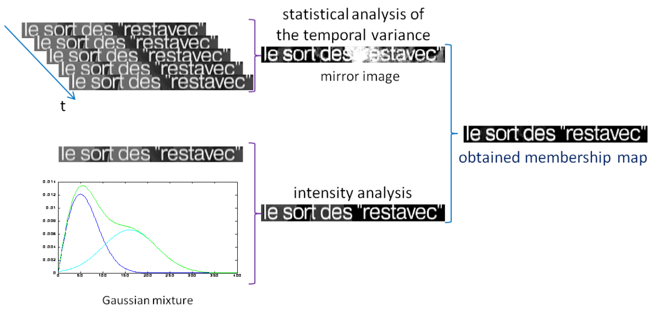

Figure 3: Fuzzy membership map generation.

3.2.2 *Shortest Path Segmentation*

We aim at finding non-linear separations between characters, well-suited to their different morphologies. Inspired by [14], where Lee et al. propose to segment texts in printed documents by using projection profiles, topographic features analysis and shortest path segmentation, we define the segmentation as a problem of shortest vertical path in the text image. Considering that each pixel, in the fuzzy map generated above, is represented by its probability to belong to the class "text", character segmentations can be identified as paths containing a sequence of pixels having a low probability (*i.e.*, a probability under a fixed threshold) to belong to the class "text". To that end, the fuzzy map is considered as a grid of vertices (pixels) and shortest paths connecting pixels from the top to the bottom without crossing the text are searched (if they exist). Paths containing pixels with an intensity value above the threshold (*i.e.*, pixels with a strong probability to belong to the "text") are discarded even if they have the lowest accumulated intensities. During path computation, three directions are allowed: 45 °, 90 °, and 135 ° with respect to the horizontal. Our purpose is to obtain separations adapted to character shapes and styles that enable an accurate recognition in the next step. In order not to compute shortest paths between every pair of vertices, and to avoid over-segmentation, a criterion of minimal distance (depending on the height of the text image) between the vertices of two successive paths is used. Finally, each obtained path is associated with the value of its costliest pixel (*i.e.*, the pixel with the hightest probability of belonging to the class "text").

Two types of segmentations are distinguished: "accurate" ones, which correspond to paths with low cost (*i.e.*, below a fixed threshold), and "risky" paths with higher cost. These "risky" ones will be questioned later. In Section 3.4, further information (recognition results and lexical knowledge) is introduced to remove ambiguities related to these segmentations. As shown in fig. 4, “risky” segmentations (drawn in red) can correspond either to an over-segmentation (e.g., the ’r’ or the ’d’) or to a separation of overlapped characters on a complex background (e.g., “rt”). We also identify paths computed between words (e.g., the segmentations between “des” and “restavec”) as areas corresponding to a space separating two words.

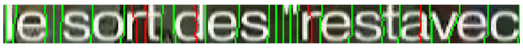

Figure 4: An example of segmentation paths: green paths are “accurate” segmentations and red ones are “risky” segmentations.

3.3 A Convolutional Neural Classification Approach for Character Recognition

In this subsection, our method to recognize segmented characters extracted from images is detailed. In the literature, a large amount of work has been dedicated to the problem of character recognition in color and in gray scale images. Most proposed OCR systems first binarize the images and then extract visual features. Using a classifier, they combine extracted information and recognize the characters. In contrast with these methods, we rely on a neural classification approach able to learn to extract appropriate descriptors and to recognize the character at the same time, and without any binarization step.

3.3.1 Convolutional Neural Network Classification

Convolutional Neural Networks, hereafter ConvNets, are a special form of multilayer perceptron proposed by LeCun et al. [13], in order to recognize visual patterns directly from the image pixels without any preprocessing step. ConvNets rely on the notions of local receptive fields, weight sharing and subsampling in the spatial domain. Due to their robustness to noise, deformations, translations and scale variations, they outperform other existing classification approaches and show a great ability to deal with a large number of extremely variable patterns.

The proposed models have been tested on many classification tasks and used for several applications [10, 18], including character recognition in color images [17] by Saïdane and Garcia.

3.3.2 Network Architecture and Training

Several network architectures were tested for our application. We finally chose a heterogeneous ConvNets configuration composed of six hidden layers and a final output layer, which proved to be very well-suited for our classification task. The proposed network avoids overfitting problems and increases the generalization ability of the system. It takes as input a normalized gray scale character image rescaled to the size of 36 × 36 pixels and returns as output a vector of N (the number of considered classes) values between -1 and 1. The recognized character corresponds to the class obtaining the highest output value. As shown in fig. 5, the first four layers consist of alternated convolutional (with a kernel mask of 5 × 5) and subsampling layers connected to three other neuron unit layers. The first two layers (a convolution followed by a subsampling layer) can be interpreted as a feature extractor. They analyze the character image and extract its low visual features (like corners or edges). 12 feature maps are then obtained and subsampled in order to reduce sensitivity to affine transformations and computation complexity. Once features are extracted, they can be combined taking into account their spatial relations. This step is ensured by the two next convolutional and subsampling layers. New feature maps are then connected to three hierarchical layers composed of classical neural units that ensure the classification decision. Outputs of the final layer encode probabilities of an input to belong to the character classes.

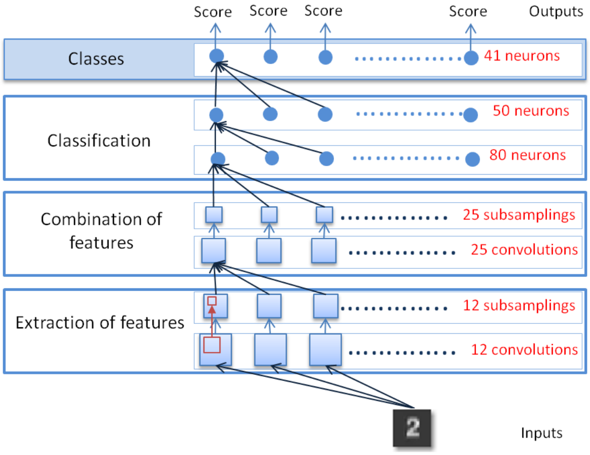

Figure 5: Convolutional neural network architecture.

The designed network was trained to learn to extract appropriate visual features and classify images of single characters. The parameters of the ConvNets (convolution coefficients, biases and connection weights) are iteratively adjusted using the error back-propagation algorithm. Classically, the database of images is divided into two sets: a training set and a validation set. At each iteration, a gray scale image of the training set is presented to the network and a mean square error term is computed as $MSE = \frac{1}{N} \cdot \sum_{i=1}^{N} (o_i - d_i)^2$, where o_i is the output value of the neuron i, and d_i is its desired value, which is set to 1 if i corresponds to the character’s class and to -1 for other neurons. The error is then back-propagated in the network layers and parameters are updated. In parallel, at each epoch[2], a classification error rate is evaluated on the training and the validation sets in order to control the generalization and the overfitting problem.

3.4 Improving Recognition Results using a Language Model

Despite its good performances (cf. Section 4), our character recognizer still generates some errors due to different reasons: similar character confusion, background complexity, segmentation errors, low video quality, etc. To tackle these difficulties and remove ambiguities associated with the local character by character recognition, we incorporate some linguistic knowledge that drives all recognition modules. Thus,

[2] An epoch is a set of iterations corresponding to the presentation of all the training images to the network.

segmentation and recognition results can be obtained taking into account both video analysis and lexical context, two completely decorrelated and complementary information.

Widely used for speech transcription and natural language processing applications like machine translation, n-gram models have demonstrated a strong ability to improve recognition performances by considering lexical context [1]. Relying on a statistical analysis of a corpus, they allow to predict the next word to be recognized given the words that have just been analyzed. In our application, a n-gram model is trained over a corpus of words in order to estimate the probabilities of sequences of letters in a given language. These probabilities are then integrated into the recognition scheme to get the most reliable word propositions.

Assuming that the goal is to find the sequence of characters $\hat{C}$ which maximizes the probability $p(C|signal)$, where C is a sequence of characters and *signal* is the given text image, we apply the Maximum A Posteriori (MAP) approach expressing the problem as follows:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, \mathrm{p}(signal|C) \cdot \mathrm{p}(C) \qquad (2)$$

where $p(signal|C)$ is the a posteriori probability of the *signal* given the recognized character sequence C, and $p(C)$ is the a priori probability of sequence C. The first term is derived from the character recognition outputs and is computed following eq. 3:

$$p(signal|C) = \prod_i p(s_i|c_i) = \prod_i \frac{\exp(output(s_i|c_i))}{\sum_j \exp(output(s_j|c_j)} \qquad (3)$$

where c_i, s_i and $output(s_i|c_i)$ are respectively the i^{th} character of sequence C, its image and its corresponding ConvNets output. The second term is obtained from the language model and can be formulated as follows:

$$p(C) = \prod_i p(c_i|\phi(h(i))) = \prod_i p(c_i|c_1 c_2 ... c_{i-1}) \qquad (4)$$

where $\phi(h(i))$ is the context of the character c_i which corresponds to the sequence of characters $c_1c_2...c_{i-1}$ that precedes it. Using the n-gram model, we assume that a character only depends on its $n-1$ predecessors. Thus eq. 4 can be rewritten into:

$$p(C) = \prod_i p(c_i|c_{i-n}c_{i-n+1}...c_{i-1}) \qquad (5)$$

Using the *SRILM* toolkit [21], the language model was trained to learn the joint probabilities of character sequences on a corpus of about 10,000 French words and named entities extracted from news texts.

Because probabilities are low, their decimal logarithm is preferred, and two coefficients γ and δ are introduced, as expressed in eq. 6:

$$\hat{C} = \arg max \sum_i (log(p(s_i|c_i)) + \gamma \cdot log(p(c_i|\phi(h_n(i)))) + \delta) \qquad (6)$$

where $\phi(h_n(i))$ corresponds to the sequence $c_{i-n}c_{i-n+1}...c_{i-1}$. γ, called the Grammar Scale Factor, encodes the weight of the language model and serves to balance the influence of the linguistic knowledge in our OCR system. The parameter δ was incorporated to compensate the over- and sub-segmentations by controlling the lengths of word candidates.

In addition to letters, numbers and punctuation marks, a class dedicated to spaces is considered in our recognition system. This class is used to identify separations between words and to treat each word separately. As shown in fig. 6, for each word, a recognition graph is built in order to determine the most probable word hypotheses that correspond to the image. For each segmented region, a series of best recognized characters is considered with their ConvNets outputs. Each segmentation hypothesis is represented by a node where a set of optimal character sequences is generated and characterized by a score that combines recognition results, language model probabilities and segmentation hypotheses. The best word candidates are determined using the Viterbi algorithm. Finally, obtained word propositions are checked using a dictionary. If no proposition belongs to the dictionary, the word with the highest score is considered as being recognized.

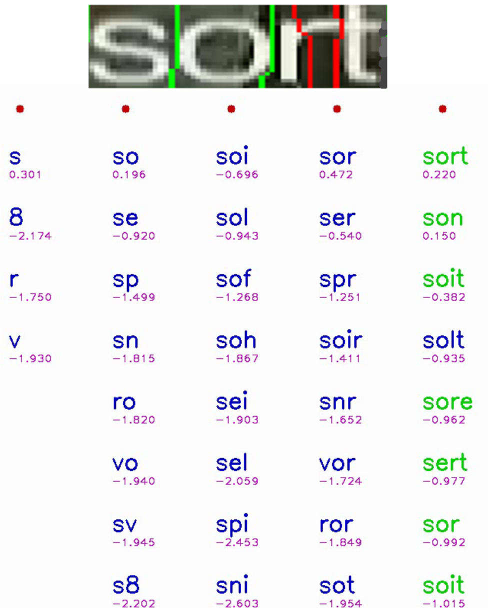

Figure 6: An example of recognition graph (words in green are words in the dictionary).

4. EXPERIMENTAL RESULTS

This section reports several experimentations and results. It starts with the description of our experimental dataset. Then, an evaluation of our neural approach for character recognition is provided. A comparison of this convolutional neural classification with a SVM-based method, in the case of our application, is also described. Finally, the impact of the language model in our complete OCR system is investigated.

4.1 Dataset

Our experiments are based on a dataset of 12 videos of French news broadcast programs. Each video, encoded by MPEG-4 (H. 264) format at 720 × 576 resolution, is about 30 minutes long, and contains about 400 words, roughly corresponding to 2,200 characters (i.e., small and capital letters, numbers and punctuation marks). As shown in fig. 7, embedded texts are strongly variable in terms of size (a height of 8 to 24 pixels), color, style and background (namely uniform and complex backgrounds).

Four videos were used to generate a dataset of images of single characters perfectly segmented. This database, used for training the neural network, consists of 15,168 character images. The other eight videos are annotated and used to test the complete OCR scheme. Single character recognition tests are first described; then the full scheme is evaluated.

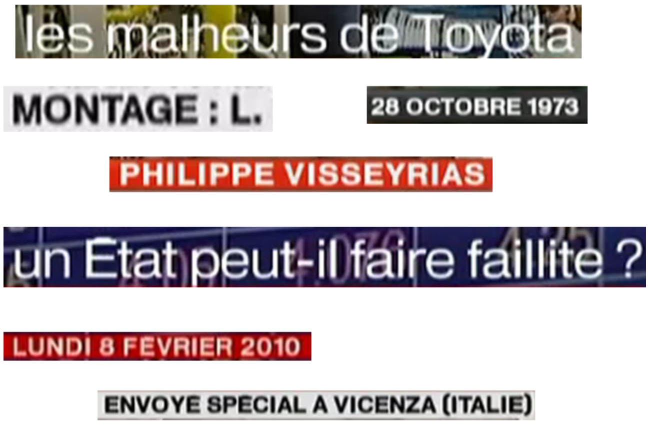

Figure 7: Samples of embedded texts in videos of news.

4.2 Recognition results

For our experiments, 41 character classes have been considered: 26 Latin letters, 10 Arabic numbers, 4 special characters (namely '.', '-', '(', and ')'), and a class for spaces between words. The dataset of single characters generated from the four videos is divided into three subsets: a training set containing 80% of the images, a validation set and a test set each containing 10% of the images. The first two sets are used to train the model (as explained in Subsection 3.3.2, two sets are necessary to control the training stage to avoid overfitting) while the last one serves to evaluate the recognition performances.

We also want to compare SVM and ConvNets classification performances for the character recognition task. Dorai et al. [9] indeed tested the ability of SVM models to recognize characters embedded in videos, but evaluated their performances on a set of real video data different from our own. Hence, to provide a meaningful comparison, both classification approaches were tested on the same database of images, generated as previouly mentioned, and under the same conditions (without extracting features, the images are considered as inputs).

Our SVM implementation was based on the software package *LIBSVM* [3], which was adapted to our dataset. Using the RBF (Radial Basis Function) kernel, several configurations of the SVM were experimented, with different values for the penalty parameter[3] C. Table 1 depicts experimented SVM models and their results.

Regarding ConvNets, several architectures were also tested on the same dataset. Details and results of some configurations are summarized in table 2. Among the tested network architectures, ConvNets_2 obtains the best recognition rate that exceeds 98%. ConvNet_1 and ConvNet_3 are also efficient but with recognition rates lower than 90%. This can be explained by a problem of underfitting for ConvNet_1 and overfitting for ConvNet_3. Indeed, the number of trainable parameters of ConvNet_1 is not sufficient to learn to classify all character classes and to remove ambiguities between similar letters such as 'I' and 'l'. ConvNet_3 is too complex with too many parameters, and the network starts to memorize images of the training set instead of learning to generalize.

Table 1: Recognition rates of SVMs on a dataset of single characters (C is the penalty term, SV means Support Vectors and RR is the recognition rate).

SVM Id	C	Number of SV	Character RR
SVM_1	1	9,215	75.46%
SVM_2	2	8,544	81.18%
SVM_3	3	8,091	80.65%

Table 2: Recognition rates of ConvNets on a dataset of single characters: C1 and C2 (resp. N1 and N2) correspond to the numbers of feature maps (resp. neural units) of the layer 1 and layer 3 (resp. layers 5 and 6).

ConvNets Id	C1	C2	N1	N2	Character RR
ConvNets_1	10	15	60	40	87.17%
ConvNets_2	12	25	80	50	98.04%
ConvNets_3	15	20	120	80	89.96%

We can see that recognition rates of ConvNets are higher than the SVM ones. Indeed, the highest ConvNets recognition rate is 98.04% (ConvNets_2 architecture). In comparison, the highest rate obtained with SVMs is 81.65% (SVM_2 model). Furthermore, in terms of trainable parameters, ConvNets_2 requires 16,732 parameters, while SVM_2 requires 8,544 Support Vectors with 11,073,024 stored parameters. Hence, we can conclude that, for this application, ConvNets yield better classification performances with lower complexity than SVMs.

4.3 Language model integration

We now focus on the evaluation of our complete OCR scheme on the eight news videos previously annotated. Since the language model is incorporated to drive the different steps of the recognition system (namely the segmentation and the recognition results), we evaluate its influence on the performances of the whole OCR system.

In eq. 6, two parameters γ and δ were introduced, that control the relative importance of the language model and the recognition and segmentation steps. Fig. 8 illustrates the evolution of the word recognition rate versus the variation of these parameters. Generally, the recognition rate tends to increase with these parameters (except when δ is higher than 3 promoting over-segmentations; red parts of fig. 8). Most obtained word recognition rates exceed the recognition rate of the baseline system (with no language model: $\gamma = 0$ and $\delta = 0$). The best system produces a word recognition rate of 74.34% with $\gamma = 0.4$ and $\delta = 2.5$.

In order to evaluate the influence of the parameter n (the length of the character history) in eq 5, bigram, trigram and quadrigram models were experimented. As shown in table 3,

[3] The penalty parameter of the SVM permits to control the trade off between allowing training errors and forcing rigid margins.

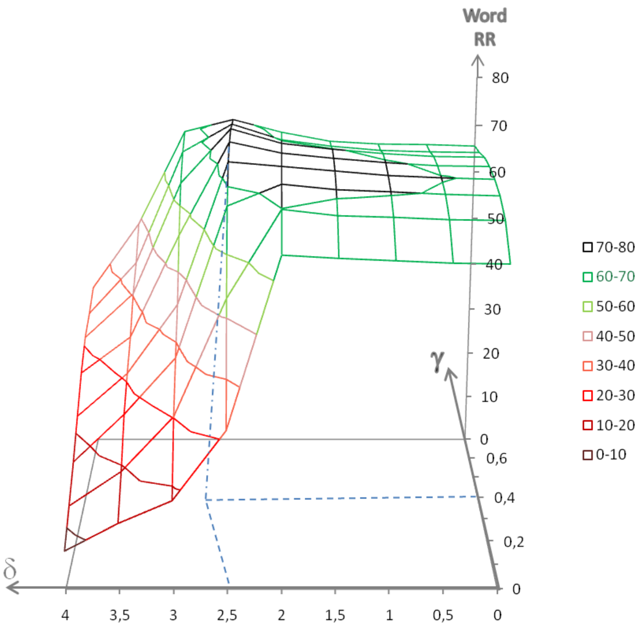

Figure 8: Influence of the weight of the language model (trigram model) on the recognition scheme (RR is the word recognition rate).

the best word recognition rate is obtained with the trigram model. The poor context (only one predecessor) considered by the bigram model, even if it improves the recognition results, is not sufficient. The quadrigram did not improve results compared to the trigram which is less complex.

Table 3: Evaluation of the influence of the length of the character history considered by the language model.

n-gram	Character RR	Word RR
Baseline system	88.14%	63.04%
Bigram	89.37%	65.45%
Trigram	92.69%	74.34%
Quadrigram	90.13%	68.78%

Fig. 9 presents some corrections due to the integration of the language model (trigram); some confusions between similar characters (e.g., 'i' and 'j') are removed, and over-segmentations (e.g., for 'r') are eliminated. However, errors associated with "accurate segmentations" (e.g., the space between 'T' and 'o' in the word "Toyota") remain difficult to correct and generally the system fails to find the right word.

Finally, we evaluated the impact of the use of a dictionary—another kind of linguistic knowledge—in the decision step. Obviously, experiments demonstrated that our recognition system can be improved when a dictionary is used, either with or without a language model (cf. table 4). They also showed that the language model has a greater positive impact on the improvement of the system recognition rate, while the combination of both sources of linguistic knowledge results in the best performances, yielding a character recognition rate of 94.95% and a word recognition rate of 78.24%. The decrease of the character recognition rate (from 98.04% to 94.95%) when testing the complete scheme can be justified by some errors of the segmentation step, of course absent in the tests on the single character dataset in Subsection 4.2; neither the language model nor the dictionary could correct them. These errors are either false segmentations considered as "accurate" ones that over-segment characters, or confusing "risky" segmentations (e.g., the word "des" that can be recognized as "cles" while remaining linguistically correct, or the over-segmentation of the letter 'd' which is not detected as an error).

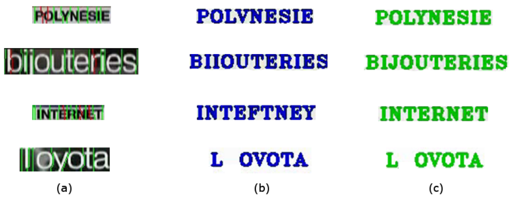

Figure 9: Examples of extracted texts recognized by the OCR: (a) images extracted and segmented, (b) results before integrating the language model and (c) results after the integration of the language model (trigram).

Table 4: Improving recognition performances by using a dictionary (LM: language model; Dic: dictionary; RR: recognition rate).

method	Character RR	Word RR
OCR	88.14%	63.04%
OCR+LM	92.69%	74.34%
OCR+Dic	90.02%	70.55%
OCR+LM+Dic	94.95%	78.24%

5. CONCLUSION AND FUTURE WORK

In this paper, we have presented a comprehensive video OCR system specifically designed to detect texts embedded in videos, to segment their images into single characters and to recognize them. Based on a neural approach, the proposed character recognizer provides outstanding results and considerably outperforms methods relying on SVM models (98.04% compared to 81.18% of character recognition rate). We have also demonstrated that the performance of our system can be improved by integrating linguistic knowledge (namely a language model) that takes into account the lexical context. This knowledge allows to reduce segmentation errors and recognition ambiguities. The proposed video OCR system was tested on a dataset of real video data and obtained very good results of about 95% of character recognition rate, corresponding to 78% of words correctly recognized.

These promising results enable the incorporation of our video OCR into an automatic video indexing and retrieval system. More precisely, the recognized texts can be used to obtain high semantic level information and index the videos. Among these information, named entities (including names of persons, of places, etc.), very common in news documents and quite evolutionary, can be acquired. Once phonetized, they can be integrated into an automatic speech recognition system, and used to extend the indexing of the speech itself. The use of our video OCR can thus improve the efficiency and the reliability of a multimedia indexing system, handling multimodal information (speech, objects, etc.). Many applications can be conceivable for this system, especially where an extension of the present work is established including the recognition of scene texts like for instance in teaching videos, presentations or conferences.

6. ACKNOWLEDGMENTS

The authors would like to thank Franck Mamalet (Orange Labs R&D) for his help, especially when developing the annotation software, and Guillaume Gravier (CNRS) for his contribution while integrating the language model.

7. REFERENCES

[1] L. Bahl, P. Brown, P. de Souza, and R. Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(7):1001–1008, 2002.

[2] R. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):690–706, 2002.

[3] C. Chang and C. Lin. LIBSVM: a library for support vector machines. 2001.

[4] H. Chang, S. Sull, and S. Lee. Efficient video indexing scheme for content-based retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1269–1279, 2002.

[5] Y. Chang, W. Zeng, I. Kamel, and R. Alonso. Integrated image and speech analysis for content-based video indexing. In *IEEE International Conference on Multimedia Computing and Systems*, pages 306–313, 2002.

[6] D. Chen, J. Odobez, and H. Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition*, 37(3):595–608, 2004.

[7] T. Chen, D. Ghosh, and S. Ranganath. Video-text extraction and recognition. In *IEEE Region 10 Conference, TENCON'04*, volume 1, pages 319–322, 2005.

[8] M. Delakis and C. Garcia. Text detection with convolutional neural networks. In *International Conference on Computer Vision Theory and Applications*, volume 2, pages 290–294, 2008.

[9] C. Dorai, H. Aradhye, and J.-C. Shim. End-to-end video text recognition for multimedia content analysis. In *IEEE International Conference on Multimedia and Expo*, pages 601–604. IEEE Computer Society, 2001.

[10] C. Garcia and M. Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, 2004.

[11] X. Hua, P. Yin, and H. Zhang. Efficient video text recognition using multiple frame integration. In *International Conference on Image Processing*, volume 2, pages 397–400, 2002.

[12] S. Kopf, T. Haenselmann, and W. Effelsberg. *Robust character recognition in low-resolution images and videos*. Universitat Mannheim/Institut fur Informatik, 2005.

[13] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, pages 255–258, 1995.

[14] S. Lee, D. Lee, and H. Park. A new methodology for gray-scale character segmentation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1045–1050, 2002.

[15] R. Lienhart and F. Stuber. Automatic text recognition in digital videos. *Image and Video Processing*, pages 2666–2675, 1996.

[16] F. Manerba, J. Benois-Pineau, R. Leonardi, and B. Mansencal. Multiple moving object detection for fast video content description in compressed domain. *Journal on Advances in Signal Processing*, 2008:5, 2008.

[17] Z. Saïdane and C. Garcia. Automatic scene text recognition using a convolutional neural network. In *International Workshop on Camera-Based Document Analysis and Recognition*, pages 100–106, 2007.

[18] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *International Conference on Document Analysis and Recognition*, volume 2, pages 958–963, 2003.

[19] C. Snoek and M. Worring. Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4):638–647, 2005.

[20] T. Som, D. Can, and M. Saraclar. HMM-based sliding video text recognition for Turkish broadcast news. In *International Symposium on Computer and Information Sciences*, pages 475–479, 2009.

[21] A. Stolcke. SRILM-an extensible language modeling toolkit. In *International Conference on Spoken Language Processing*, volume 3, pages 901–904, 2002.

[22] R. Yager. Connectives and quantifiers in fuzzy sets. *Fuzzy sets and systems*, 40(1):39–75, 1991.

[23] J. Yi, Y. Peng, and J. Xiao. Using multiple frame integration for the text recognition of video. In *International Conference on Document Analysis and Recognition*, pages 71–75, 2009.

[24] D. Zhang and S. Chang. A Bayesian framework for fusing multiple word knowledge models in videotext recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 528–533, 2003.