Transcription of manuscrits with image processing techniques and gamesourcing

Jialuo Chen

Abstract– Information inside the historical documents can provide us knowledge about the evolution of the past. In local censuses, there are names that appear the 80% of times. The transcription process could be accelerated doing a massive transcription of frequent names. In this work we propose to use clustering methods and validate them via gamesourcing. The validation is needed because the performance of image processing techniques is still far from satisfactory. Several experiments are performed showing the viability of the massive transcription through clustering methods and the gamesourcing application for validation.

Keywords– Crowdsourcing, Gamesourcing, Massive transcription, Clustering, Segmentation, Label propagation, Hierarchical K-Means

1 Introduction

AROUND the world, there are millions of historical documents, but only the 10% or less of them are digitized, and from them, a minimum amount are transcribed or indexed.

Population sources contain information of our ancestors. They allow the study of the demographic behaviour, the migratory waves and the understanding of the social and economic evolution of the past. The aim of the research project XARXES[1] is to develop technologies to create historical social networks based on the linkage of citizens registered in the local census from neighbouring municipalities.

Therefore, instead of using an big amount of human resources to extract and read the information, the idea is use document analysis techniques to automatically process the information contained in these documents.

The first step to construct the social network is to extract the information contained in the census records. Currently, there are many techniques to recognize handwritten texts, but their performance is still far from satisfactory when dealing with historical manuscripts. For reason of document degradation and different writing styles, a manual validation is mandatory.

In local censuses, scholars estimate that there is a 20% of names and surnames that appear the 80% of times. Hence, the hypothesis is that the transcription of these manuscripts could be accelerated through the detection and transcription of these frequent names. Afterwards, the transcriptions could be manually validated via crowdsourcing [2]. The idea of crowdsourcing is to split the work in many small tasks and solicit the contribution from people, specially from the online community. Wikipedia [2] is a perfect example of crowdsourcing with thousands of contributors.

Despite the use of, the validation of transcriptions takes a long of time and it can be very boring for the volunteers. Gamification, defined as the application of game-design elements and principles in non-game contexts, has demonstrated to engage and keep the interest of users. Lately, it has been also applied to crowdsourcing activities [16], such as the $Digitalkoot$ [7] transcription games at $Facebook$. Therefore, users could be more engaged in the validation thanks to gamesourcing (understood as crowdsourcing via gamification), and consequently, the massive transcription and validation of these documents can be accelerated.

The rest of the paper is organized as follows. Section 2 describes the objectives of this work. In the Section 4 shows the system architecture of the gamesourcing. The algorithms used to analyze and process the images are explained in the Section 5 and 6. Results are shown in the Section 7. Finally, Section 8 shows the conclusion and future work.

2 Objectives

The objective of this work is to perform a massive transcription of these frequent names and surnames via word clustering, and implement an Android application to validate these transcriptions via gamesourcing. The sub-objectives are:

- Create a segmentation algorithm that can analyze a full page of historical document and segment all the words.

• E-mail de contacte: jialuo.chen@e-campus.uab.cat
• Mención realizada: Enginyeria de Computació
• Treball tutoritzat per: Alicia Fornés, Pau Riba (Ciències de la Computació)
• Curs 2017/18

[1] http://dag.cvc.uab.es/xarxes/

[2] https://www.wikipedia.org/

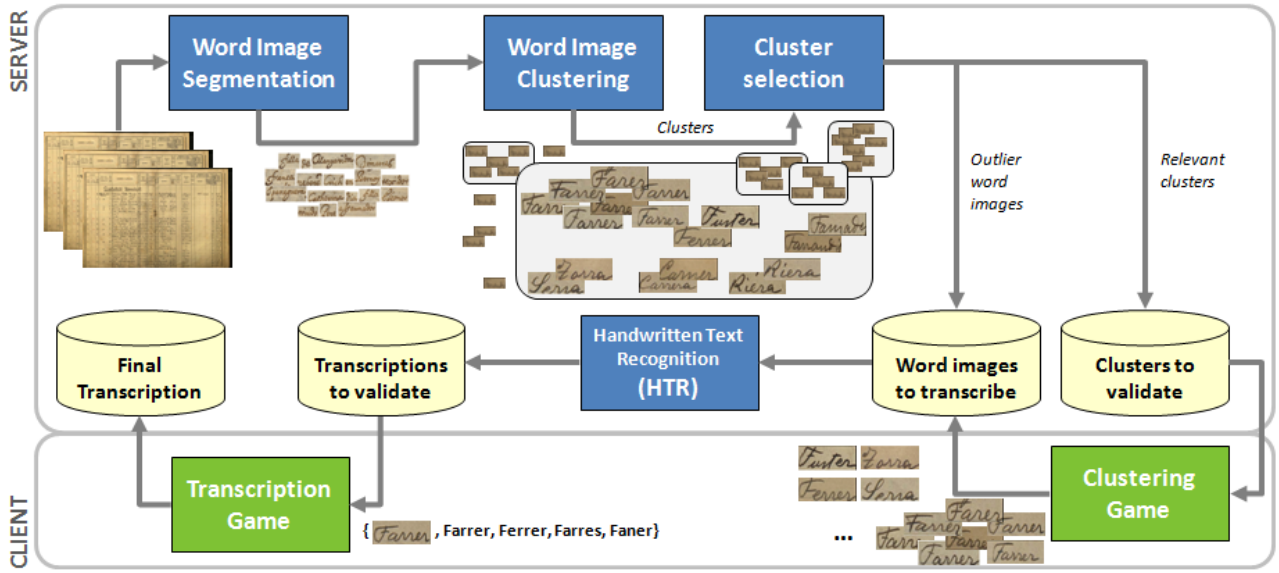

Fig. 1: System architecture. The server feeds the Android games with images, and analyzes the user's feedback in order to validate the transcriptions.

- Develop a clustering algorithm that can group those segmented images.
- Combine segmented word images with synthetic words to perform the clustering and evaluate if it can be transcribed without a HTR.
- Create an android application prototype that can manage the gamesourcing.
- Experimentally analyze these algorithms and study the performance of the android gamesourcing application.

3 State of the Art

Handwritten Text Recognition (HTR) has lately received much attention. Actually, HTR is based in Hidden Markov Models (HMM), Recurrent Neural Networks or combinations of different techniques. For instance Toledo *et. al.* [18] have recently proposed a technique with the combination of Pyramidal Histogram of Characters (PHOC), Convolutional Neural Networks (CNN) and Bi-directional Long Short-Term Recurrent Neural Networks (BLSTM-RNN) to transcribe segmented word images.

Due to the nature of handwritten words, the transcription is not perfect, especially for historical manuscripts. Hence a manual validation is needed. In this paper [9] they presented a crowdsourcing web-based application to extract information from demographic handwritten document images. To perform a massive transcription of frequent names, a clustering method is necessary. Therefore, unsupervised clustering like K-Means [11] has been used. Because the input of clustering algorithm are segmented word images, segmentation techniques is proposed. In the area of segmentation techniques, there are many variety of segmentation methods. Threshold segmentation [1] being the simplest method, or techniques with more complexity like segmentation based in weakly-supervised learning in CNN [5].

4 System Architecture

The complete system architecture is shown in Figure 1. The main components of the system are hosted in a server, while the gamesourcing apps run in an Android client. Given a collection of handwritten documents to transcribe, word images are segmented. Then, word images are clustered to find high frequency words that can be jointly transcribed. The clusters are validated using the first of the proposed gamesourcing applications, the clustering game. The HTR module generates plausible transcriptions of these word images. The second gamesourcing app, the transcription game, is used to validate the transcriptions.

4.1 Image processing

Given that the games need segmented word images, the first step consists of a segmentation algorithm that extracts all the words from a collection of handwritten documents. Given that the objective is to transcribe frequent words, a clustering algorithm is needed to find high frequency words to be jointly transcribed. A cluster selector will discard outlier word images (small clusters, isolated instances). These clusters are validated by one of the gamesourcing application, the clustering game.

Validated cluster images and discarded word images (those images that do not belong to any cluster, or images discarded by the clustering game) will be transcribed together. The HTR generates probable transcriptions of these word images. These transcriptions and the corresponding word image are sent to be validated by the second gamesourcing application, the transcription game.

4.2 Android application

The android application of the client part has the following functionalities:

- Two mini-games with their corresponding play instructions: Transcription game and Clustering game,

that inside the game they are called Match game and Difference game (for more information, see the Appendix Section A.1). Both games use segmented word images to play.

- User account: create a new account, log in and log out.
- The TOP 10 scores of the both games.
- Language selection. This prototype has four different languages to display: Catalan, Spanish, English and Chinese.

In the clustering game, it shows images of a given group, and the user is asked if those words are the same. For the transcription game, it shows the word image with the corresponding plausible transcriptions, and the user has to select the correct transcription (if there is not a correct transcription, they have a button to say it). When the game finishes, these validations is sent back to the server and processed.

For the android game, we have a automatic update of images that synchronizes with the database. This automatic update actives when all the word images located in the client part have been validated. It downloads new images and erase unused images, optimizing the client storage.

Since this application need to download images and send users feedback, the interaction with the database is done with a PHP server that controls the creation of new users, the transcription and clustering results. We use a MySql server to define the database.

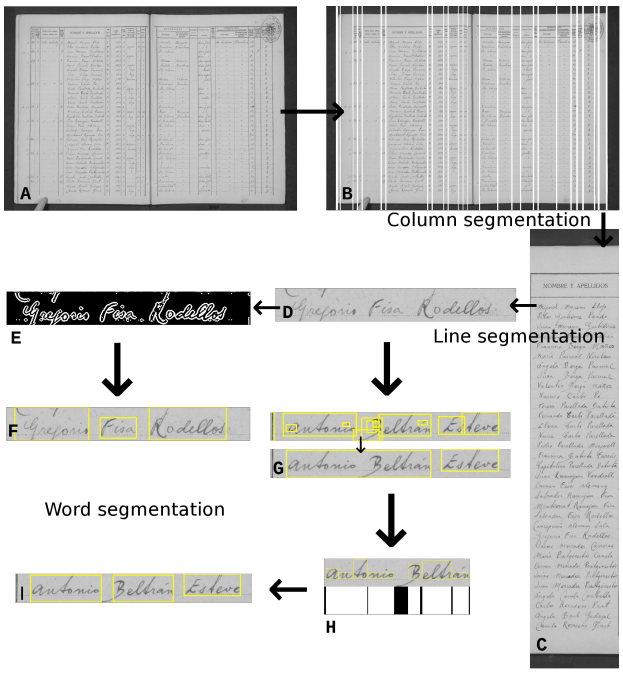

Fig. 2: The pipeline of the segmentation algorithm (census document from 1940). (A) Full page of a local census. (B) Hough transforms applied in the full page. (C) Segmented column. (D) Segmented word line. (E) Anisotropic Gaussian Filter applied to the text line. (F) Detected words in bounding boxes. (G) An example of bad segmented word line. (H) Trouble section of the word line with it binary mask. (I) Word line with bounding boxes.

5 Segmentation Algorithm

Given that the census documents correspond to full pages with names, ages, occupations, and the addresses where citizens lived, we first need to extract the words before the clustering process. The segmentation algorithm has a big impact in the entire system. With good segmented images, the accuracy of the clustering process will improve. In this work, we will use table based historical document collections.

The pipeline of the segmentation algorithm shown in the Figure 2 (for images at higher resolution see the Appendix A.2). We can divide the algorithm in three sections: Column, line and word segmentation.

For the column segmentation, we have the full page of the document (A). First, using Hough transform we get the vertical lines that is shown in the part (B). Then, using the given lines we segment the desired column. In the line segmentation, we have the column segmented in the last step (C). Using morphology and projections we can segment these lines (D). In normal conditions, we have a word line without much noise that the bounding boxes of the words can be easily find (E). But if we have a word line with noise, then we need apply optional steps (G, H) to reach to the bounding boxes of the words (I). We will explain every step in detail next.

5.1 Column

To know the position of the column (in our case the column of first names and last names), we need to know first where are the vertical lines that divide the document in columns.

Using the Hough transform algorithm we obtain segments of vertical lines of the table, and then we filter them grouping those lines that are in the same x position. After the line grouping, for each group we draw one line from the first line x position to the last line x position, getting a image which we know where are the columns. Now we can get the column of names that we are interested with some conditions. These conditions can change depending the document. In this case, the column of names is the largest one.

5.2 Line

Segmenting by row the column we can get lines with words (for each line we have the first name and two last names). First we projected the image in binary by rows and search local maximums. Where we have a maximum, there is a high probability that we have a word line. And then, for each maximum we filter it by height and pixel ratio. These segmented images that pass the filter will be the word lines.

5.3 Word

Applying Anisotropic gaussian filter [8] we can know where are the words. This filter process the input image and return the same image but binarized with the words and strokes surrounded with white pixels. Figure 2 section (E) shows the output of the anisotropic gaussian filter. Hence, it is easy to find where are the words. Finally, using connected components we can detect the group of pixels and finally get the bounding boxes.

However, we cannot correctly segment all the words. Those words with long strokes in the capital letter that are too close to another word makes the algorithm segment them like one word. Figure 2 section G shows an example of a bad segmented word line that in the following steps will be corrected.

In order to avoid clustering bad segmentated words, first we need to detect them. Checking the width of the segmented words, we can decide if is a good segmentation or not. To solve the bad segmentation, we use again the anisotropic gaussian filter to find the bounding boxes. But this time, our input image is cropped and only the middle part (horizontally) is passed. Hence, applying connected-components to the resulted binary image will only have bounding boxes where the letters are. Doing a projection of the resulted image will serve us to know where we can segment.

We can see in the mask that we have many black lines. These black lines indicates the space where we do not have letters. We find those black lines (in this case, one) that can provide the right position of the segmentation filtering them with the median width. Those black lines that are smaller or equal than the median width are discarded. And those lines that starts in the beginning of the image or in the end of the image are discarded too.

Those remaining lines are ordered by the closest to the furthest from the image center. And finally, we can segment the words using the given position by the lines.

6 CLUSTERING

To make the massive transcription of frequent names possible, we need to cluster all these segmented images that are similar between them. We will explore two different types of clustering to see what works better, the unsupervised clustering and the semi-supervised clustering.

6.1 Unsupervised clustering

The unsupervised clustering is made up a K-Means algorithm with two stages:

1. Cluster gray-scale images by aspect ratio.
2. Hierarchical clustering with dense SIFT descriptors [12] from images clustered by aspect ratio.

In the first stage, we cluster the images in three classes: small, normal and large. This process will make the hierarchical clustering avoid those groupings with different sizes of images.

In the second stage, we do a hierarchical clustering with three parameters [15] to control the output clusters. These three parameters are:

$$WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2 \quad (1)$$

$$BSS = \sum_{i} |C_i| \, (m - m_i)^2 \quad (2)$$

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \quad (3)$$

- Average compactness (in K-Means distances space): Metric that indicates the cohesion of the clusters. Equation 1 shows the Within Sum of Squares (Cluster cohesion). x is a sample of cluster i and mi is the centroid of this cluster..
- Average separability (in K-Means distances space): Metric that indicates the separation of the clusters. Figure 2 shows the Between Sum of Squares (Cluster separability).m is the global centroid and mi is the centroid of the cluster i Ci is the size of the cluster i.
- average silhouette score [17] (score ranges from -1 to +1): Metric used in classification algorithms that measures the similarity of the sample with the cluster and the neighboring clusters. If the score is nearby to -1, it indicates a clustering configuration with too few or too many clusters. If the score is near to 0, it indicates that the samples are too similar. And if the score is close to 1, it indicates an appropriate clustering configuration. Figure 3 shows the Silhouette score. Being *a(i)* the measure of how well *i* is assigned in it cluster and *b(i)* the lowest average distance of *i* to all points in any other cluster, where *i* is not a member.

Before the clustering, we have to compute SIFT descriptors of the input images. In the deeper levels of the clustering, the grid size of the SIFT descriptor increases (the SIFT feature vector is longer) so that we can highlight small differences between the words.

We have two forms to initialize the centroids of the K-Means algorithm. The first one is completely automatic where the algorithm chooses randomly the centroids. And then, we do the clustering process many times varying the number of centroids (the variation is calculated with the number of samples we have to cluster).

The resulted clusters are groups with the minimum compactness, maximum separability and the maximum silhouette score, avoiding clusters of one or two images (those clusters with few images / discarded images are sent directly to the HTR).

And the second form is semi-automatic, where the centroids are chosen randomly too but we choose for each class, n centroids. Even the centroids are chosen randomly, they will be always the same to compare the results of different clustering algorithms. And then, we do the clustering process but only one time, because the number of centroids are always the same and the centroids too. In deeper levels of clustering, if we have a group of images that does not have previously selected centroids, will be discarded. So, in this initialization then number of avoided clusters will be bigger than the previous configuration.

For this configuration we will ignore the metrics to choose the optimal clusters. Since the clustering process will be done only one time.

6.2 Semi-supervised clustering

Semi-supervised clustering can be useful too to cluster word images. But since these algorithms need training data (because in the training step it is completely supervised), first of all we need some useful data to train it.

For the training data, we can use the clustering method explained before to get groups of images. For each group, we compare them in pair to see the similarity. Only groups with high similarity (90% accuracy) can be used to train the semi-supervised algorithm. With these high similarity groups of images, we can fit the algorithm.

The algorithm we used for the semi-clustering process is the label propagation [14]. This algorithm can be found at the Sklearn library [13].

The kernel for the label propagation is the core algorithm to propagate the labels. We tested two algorithms, the knn (K-nearest neighbors algorithm which creates a graph that connects the input samples to the nearest n neighbors) and the rbf (Radial basis function algorithm that depends on the distance from the labeled one to the non-labeled one).

The input for this algorithm are SIFT features [12] of segmented images and their labels if they are labeled images. And the output of the algorithm can be hard-assignment labels or soft-assignment probabilities.

Depending on the used kernel, the output groups will change. For example, in the Figure 3 we can see the same data but with two different configurations.

In the rbf configuration, the propagation method is based in the similarity measure between the labeled and the unlabeled data. In the case of the data 2 and 3, their label will be B. And the same for the data 4, although it can receive the label information from A, the data 4 is closer to B. For the data 1, we decided not to label it. Although it is close enough to receive the label information from A, it's still far from being labeled like A.

In the knn configuration, the propagation method is based on the number of labeled neighbours it has. For the data 1, the label will be A. For the data 2 and 3, the label will be B. But for the data 4, as it has one neighbour of A and one of B, so we decide not to label it.

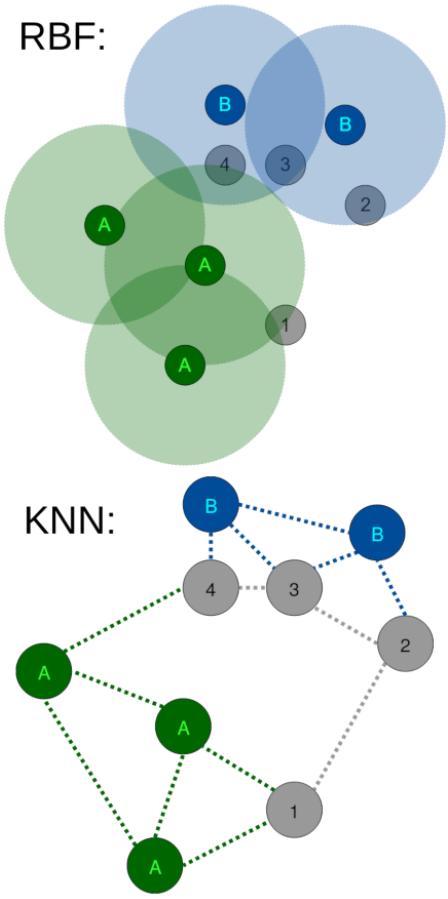

Fig. 3: Example of resulting clustering when using different kernels.

7 Results

Given the large amount of types of handwritten census documents, we selected a table based collection for our project. The input for the segmentation algorithm are full census pages with tabular form, concretely from the year 1940. As we aim to do a massive transcription of frequent names, which are essentially first names and last names, we are interested in those columns with first and last names. As we need to compare both clustering algorithms accuracy, we need data with its groundtruth. So we have used the training set of the ICDAR-IEHHR competition [10], where words have been already segmented, with their groundtruth, and they also have the same handwriting style (2.854 images in total). For the transcription accuracy we used the WER [6] (Word Error Rate) to show the results.

7.1 Segmentation

Most of the segmented words are like the examples we explained in the Section 3. As we can see in Figure 2 section C, the segmented column shows lines of words that contain spaces between one word to another. But in Figure 4, we can see the word *Montquillot* and *Campderros* are very close, so the algorithm cannot separate them. Although we have the segmentation method that is robust to noise, we cannot segment it if we do not know where are the positions to segment (In the Figure 5 we can see that in the mask the right line to segment the image did not appeared).

Another type of words where the algorithm fails are words with light pressure (in the Figure 6, the word *Vileprat* has a low pressure in the beginning of the letter *r*). Vanishing strokes caused by light pressure affects the binarization process of the image, making the anisotropic gaussian filter divide the word and getting bad bounding boxes when applying connected components.

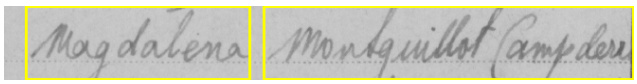

Fig. 4: Example 1 of bad segmented words.

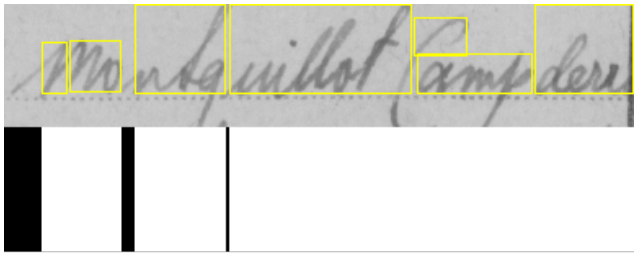

Fig. 5: Steps of the robust segmentation of the example 1 of bad segmented words.

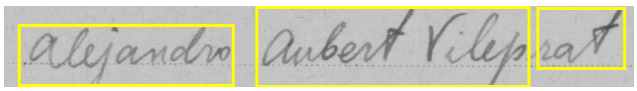

Fig. 6: Example 2 of bad segmented words.

Extra information of the segmentation algorithm can be found in the appendix, section A.2.

7.2 Unsupervised clustering

We test our algorithm with two configurations. The first configuration will be completely automatic, with random initialization for the centroids. This configuration will test the algorithm in front of situations that where the groundtruth is not available or we have few labeled samples.

And the second configuration will be semi-automatic, with be initialized with manually chosen seeds. This configuration will show the differences between random initialization and manually initialization.

Apart from these two configurations, we generated synthetic words for each transcription/class. In this way, we can see if synthetic words can help us improving the clustering. The aim is to transcribe handwritten images clustering the synthetic words with the train set, instead of using a HTR method. Figure 7 shows the difference between the segmented word image and the generated synthetic word.

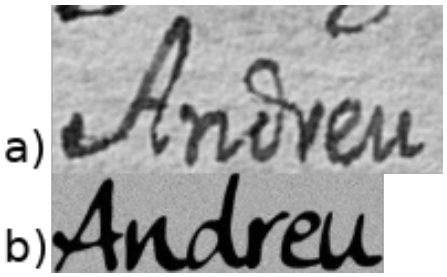

Fig. 7: a) Segmented word image. b) Synthetic word image.

Metrics to validate the algorithm:

- Cluster accuracy (Clusters with and without synthetic words): If the images in the cluster are the same, this percentage increases. We will divide the wrong cluster rate in four categories:
  - Totally wrong.
  - Only one image is different.
  - Only two images are different.
  - Only three images are different.
- Transcription accuracy (only for clusters without synthetic words): Take the majority transcription of the cluster and compare it with the ground truth. If they are the same, this percentage goes up.

Table 1 shows the results of the unsupervised clustering without synthetic words. The first thing we can notice is the difference of the number of clusters. For the random initialization of centroids, we have more than 450 output clusters. Compared with the chosen initialization, they are too many.

TABLE 1: RESULTS OF THE UNSUPERVISED CLUSTERING WITHOUT SYNTHETIC WORDS.

	Random	**Manual**
Discarded images	0	569
Cluster		
Number of clusters	461	52
Most repeated number	3 images	+15 images
of images per cluster	27.55%	71.15%
Correct	75.92%	42.31%
- Totally wrong	0.87%	25.00%
- One image	16.05%	25.00%
- Two images	5.21%	3.85%
- Three images	1.95%	3.85%
Wrong (Total)	24.08%	57.70%
Transcription		
- Correct (100-WER)	93.42%	90.81%
- Wrong (WER)	6.58%	9.19%

On one hand, having amount of clusters means that most of your clusters are small. Table 1 shows that, about 30% of the clusters have only three images. Having clusters with small sizes helps to fit in the same screen of the android game, making the validation speed goes up. But after the validation, these clusters are too small to realize a massive transcription.

On the other hand, using those big clusters of the manually chosen initialization clustering, we can arrive to the same validation speed splitting them in to small clusters. And before the transcription step, we join those clusters that were once one. Thus, we can perform a massive transcription using big clusters.

Another relevant thing we can see in the Table 1 is the accuracy of transcription. Even with a small amount of clusters, the chosen initialization transcription accuracy is almost the same compared with the random initialization accuracy. That shows that the quality of the clusters is very high.

TABLE 2: RESULTS OF THE UNSUPERVISED CLUSTERING WITH SYNTHETIC WORDS.

	Random	**Manual**
Discarded synthetics	2447	2319
Cluster		
Total number of clusters	461	53
Number of clusters	53	32
with synthetics	11.50%	60.37%
Synthetic in		
the right cluster	74.58%	56.91%
Transcription		
- Correct (100-WER)	94.00%	82.82%
- Wrong (WER)	6.00%	17.18%

Table 2 shows the results of the unsupervised clustering with synthetic images. Synthetic word images were generated from the groundtruth of the training set (2.565 syn-

thetic images in total) with 57 different font styles.

We can see that almost all of synthetic words are discarded. This discarding is influenced by the font style of the synthetic words. But even with this amount of discarded synthetics, in the manually chosen initialization clustering, we have more clusters with synthetic than without. And the accuracy of them is more than 50%.

The usage of synthetic word images does not help the clustering to do it better. There are too many discarded synthetic images, and that shows unfeasible of clustering segmented words with synthetic words.

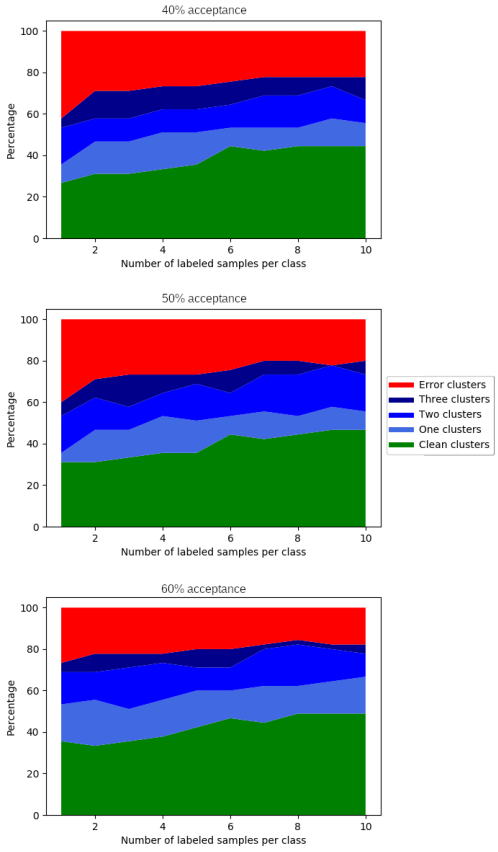

Fig. 8: Three graphics with their respective acceptance percentage changing the number of labels we choose for each class.

7.3 Semi-supervised clustering

We chosen the knn kernel for the label propagation to test it with the same configuration. The reason to choose the knn kernel is that you have more control of the connected samples than rbf kernel that uses a similarity measure.

The label propagation uses soft-assignment for the label propagation, so before the experiments with specific initialization, we analyzed the influence of the acceptance percentage when we perform the label propagation.

In Figure 8 we can see three graphics with different types of clusters. *One clusters* are clusters with only one different image. *Two clusters* are clusters with only two different images and the same with *Three clusters*. The *Error clusters* are clusters with more than three different images inside the cluster.

As we can see, using 6 or more labeled samples (chosen initialization for the label propagation) we can reach to the same clean clusters percentage. Because the more different examples we have for one label, the more confidence we have when we spread the label.

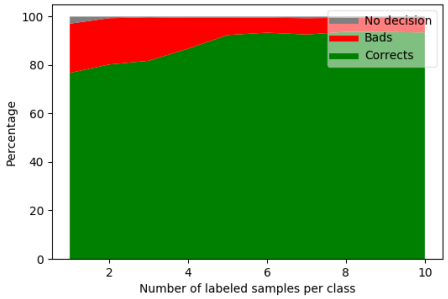

Fig. 9: Variations of the percentage of correctly transcribed, incorrectly transcribed and no decision in terms of numbers of labeled samples per class with 50% of percentage acceptance.

After deciding that the best percentage acceptance is 50% with 6 labeled samples per class, we computed the graphic of the accuracy of transcriptions that is shown in Figure 9, to verify that this decision does not influence the transcription accuracy. Or even better, if the transcription accuracy increases.

Table 3 shows the results of the label propagation with different configurations. We can see that the label propagation with random configuration gives the worst results until now. Almost all labels have been assigned probabilities lower than the acceptance percentage, discarding all word images.

Despite the random configuration, the chosen initialization has very good results with chosen initialization. Even with 55.56% of wrong clusters, the transcription accuracy is very high.

TABLE 3: RESULTS OF THE LABEL PROPAGATION WITHOUT SYNTHETIC WORDS.

	Random	**Manual**
Discarded images	2854	282
Cluster		
Number of clusters	0	45
Most repeated number	1 image	+15 images
of images per cluster	0.00%	95.55%
Correct	0.0%	44.44%
- Totally wrong	0.0%	24.44%
- One image	0.0%	8.88%
- Two images	0.0%	11.11%
- Three images	0.0%	11.11%
Wrong (Total)	0.0%	55.56%
Transcription		
- Correct (100-WER)	0.0%	93.78%
- Wrong (WER)	0.0%	6.22%

In Table 4, we show the results of label propagation with synthetic words with the manually chosen initialization. we

only tested the manual initialization, because the random one will do the same discarding all the synthetic words.

We can see that almost all of clusters have some synthetic words, although the accuracy in clusters is lower than 50%, the accuracy in transcription remains high.

TABLE 4: RESULTS OF THE LABEL PROPAGATION WITH SYNTHETIC WORDS. ALL VALUES WITH COMMA ARE BETWEEN 0-100% (WER: WORD ERROR RATE [6])

	Manual
Discarded synthetics	677
Cluster	
Total number of clusters	45
Number of clusters	42
with synthetics	93.33%
Synthetic in	
the right cluster	32.79%
Transcription	
- Correct (100-WER)	75.71%
- Wrong (WER)	24.29%

7.4 Comparison between unsupervised and semi-supervised clustering

After the comparisons of the same method with different initialization, now we will see the comparison between different methods.

TABLE 5: RESULTS OF THE LABEL PROPAGATION WITHOUT SYNTHETIC WORDS.

	K-Means	**Label Propagation**
Discarded images	569	282
Cluster		
Number of clusters	52	45
Most repeated number	+15 images	+15 images
of images per cluster	71.15%	95.55%%
Correct	42.31%	44.44%
- Totally wrong	25.00%	24.44%
- One image	25.00%	8.88%
- Two images	3.85%	11.11%
- Three images	3.85%	11.11%
Wrong (Total)	57.70%	55.56%
Transcription		
- Correct (100-WER)	90.81%	93.78%
- Wrong (WER)	9.19%	6.22%

Table 5 shows the results of both methods using manual chosen initialization without synthetic words.

The numbers are very similar, but the label propagation method is slightly higher than the K-Means method. The label propagation method discarded less word images and it had generally more images per cluster than the K-Means. Even with more wrong clusters of two and three images, the transcription accuracy is still higher than the unsupervised method.

7.5 Android game

The image database we used for the gamesourcing experience corresponds to 938 instances of surnames from the marriage records of the Barcelona Cathedral [10]. And the HTR has been trained with the training set of the ICDAR-IEHHR competition [10].

The android application was tested to see the users feedback. All of the participants that are volunteers to transcribe census documents said that the android application is better than the web page to perform a transcription.

The experiment was realized with different typology of users: foreigners, natives and experts. Analyzing the users' feedback, we saw interesting things. In the validation stage of transcriptions, foreigners disagree in which is the correct word more often than the other type of users, because to they do not know the catalan language. For further details, the reader is referred to [4].

8 CONCLUSIONS AND FUTURE WORK

In this work we have proposed the massive transcription using different techniques. We have used low complexity algorithms to perform word images clustering and, then, we have validate the transcriptions with a gamesourcing application. From the experiments, we have shown that it is completely viable and reduces significantly the number of word images to be transcribed.

However, the usage of synthetic words to avoid the HTR to transcribe needs more improvement. The generation of thousands of synthetic word images is simple, but to find the font-style that is similar to the historical manuscripts is very difficult, because of the background generation, the ink color, light pressure letters and the handwriting style.

With this work, we published one workshop [3] in the International Workshop on Document Analysis Systems (DAS) and one paper [4] in the International Conference on Frontiers in Handwriting Recognition (ICFHR).

Future work will be focused on the generation of more realistic synthetic words using neural networks. Also, we could improve the clustering algorithm with better comparison techniques in samples features that can find similarities between segmented word images and synthetic images.

Concerning the segmentation algorithm, the future work will be to solve these bad segmentations. We could improve the binarization process that keeps those pieces of letters with vanishing strokes, easing this problem.

ACKNOWLEGMENT

Thanks to all the users that participated in the gamesourcing test. Personally, I would like to thank my two tutors that helped me a lot in this project.

Thanks to my family and my friends that supported me through the work.

REFERENCES

[1] Salem Saleh Al-amri, Namdeo V. Kalyankar, and Khamitkar S. D. Image segmentation by using threshold techniques. *CoRR*, abs/1005.4020, 2010.

[2] A. Amato, A.D. Sappa, A. Fornés, F. Lumbreras, and J. Lladós. Divide and conquer: Atomizing and parallelizing a task in a mobile crowdsourcing platform. In *Int. ACM Workshop on Crowdsourcing for Multimedia (CrowdMM)*, pages 21–22, 2013.

[3] Jialuo Chen, Alicia Fornés, Joan Mas, Josep Lladós, and Joana Maria. Word-hunter: Speeding up the transcription of manuscripts via gamesourcing. In *International Workshop on Document Analysis Systems*, 2018.

[4] Jialuo Chen, Pau Riba, Alicia Fornés, Joan Mas, Josep Lladós, and Joana Maria. Word-hunter: A gamesourcing experience to validate the transcription of historical manuscripts. In *International Conference on Frontiers Handwriting Recognition*, 2018.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016.

[6] Wikipedia Community. Word error rate. https://en.wikipedia.org/wiki/Word_error_rate, last entry: 11/04/2018.

[7] DigitalKoot. Url:http://www.digitalkoot.fi/.

[8] David Fernandez, Josep Lladós, and Alicia Fornés. Handwritten word spotting in old manuscript images using a pseudo-structural descriptor organized in a hash structure, 06 2011.

[9] Alicia Fornés, Josep Lladós, Joan Mas, Joana Maria Pujades, and Anna Cabré. A bimodal crowdsourcing platform for demographic historical manuscripts. In *International Conference on Digital Access to Textual Cultural Heritage*, pages 103–108, 2014.

[10] Alicia Fornés, Veronica Romero, Arnau Baró, J Ignacio Toledo, Joan Andreu Sanchez, Enrique Vidal, and Josep Lladós. Competition on information extraction in historical handwritten records. In *ICDAR*, pages 1389–1394, 2017.

[11] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, Jul 2002.

[12] Ebrahim Karami, Mohamed S. Shehata, and Andrew J. Smith. Image identification using SIFT algorithm: Performance analysis against different image deformations. *CoRR*, abs/1710.02728, 2017.

[13] Scikit learn library for Python. Computer vision tools. http://scikit-learn.org/stable/index.html, last entry: 25/05/2018.

[14] Scikit learn library for Python. Semi-supervised. http://scikit-learn.org/stable/modules/label_propagation.html, last entry: 25/05/2018.

[15] K. M. Lee, K. M. Lee, and C. H. Lee. Statistical cluster validity indexes to consider cohesion and separation. In *2012 International conference on Fuzzy Theory and Its Applications (iFUZZY2012)*, pages 228–232, Nov 2012.

[16] Benedikt Morschheuser, Juho Hamari, and Jonna Koivisto. Gamification in crowdsourcing: a review. In *HICSS*, pages 4375–4384. IEEE, 2016.

[17] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65, 1987.

[18] J Ignacio Toledo, Sounak Dey, Alicia Fornés, and Josep Lladós. Handwriting recognition by attribute embedding and recurrent neural networks. In *International Conference on Document Analysis and Recognition*, pages 1038–1043, 2017.

APPENDIX

A.1 Android application games

Fig. 10: The transcription game called Match Game. For the selected word (top right), its possible transcriptions are shown. The user should select the transcription *Costa*.

Fig. 11: The clustering game called Difference Game. The player has to validate the correctness of the cluster and remove possible outliers. In this example, *Poch* does not belong to this cluster *Pons*, so the user has to select it.

A.2 Segmentation results

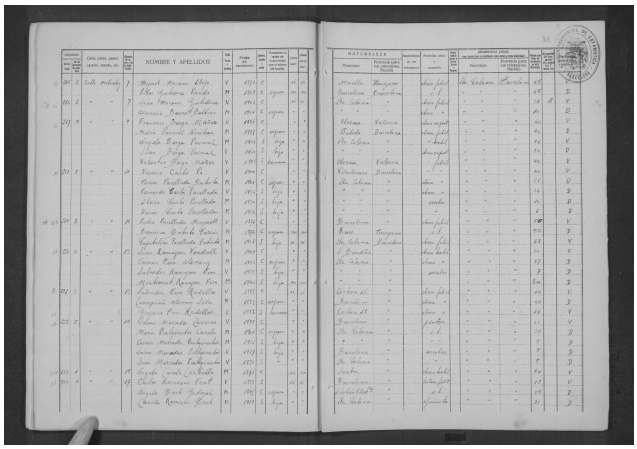

Fig. 12: Page of Census of 1940.

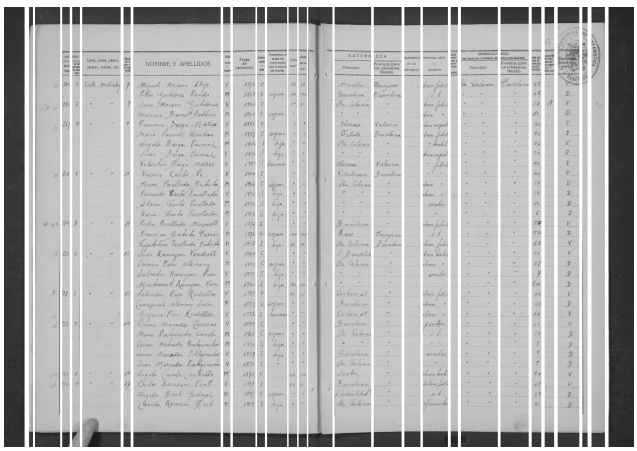

Fig. 13: The Census page after drawing these lines find by the Hough transform algorithm.

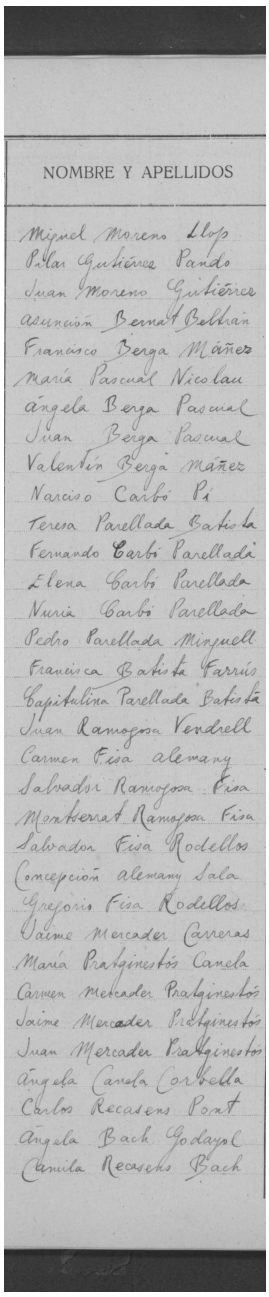

Fig. 14: The segmented column of first names and last names.

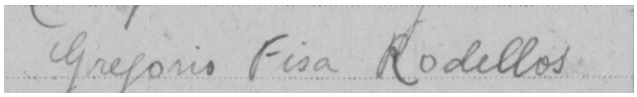

Fig. 15: Segmented line with the first name and last names.

Fig. 16: Result of the anisotropic gaussian filter [8].

A.3 Workplan

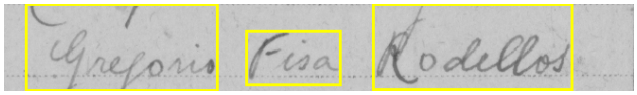

Fig. 17: The line with bounding boxes after apply the anisotropic gaussian filter [8] algorithm and found the bounding boxes.

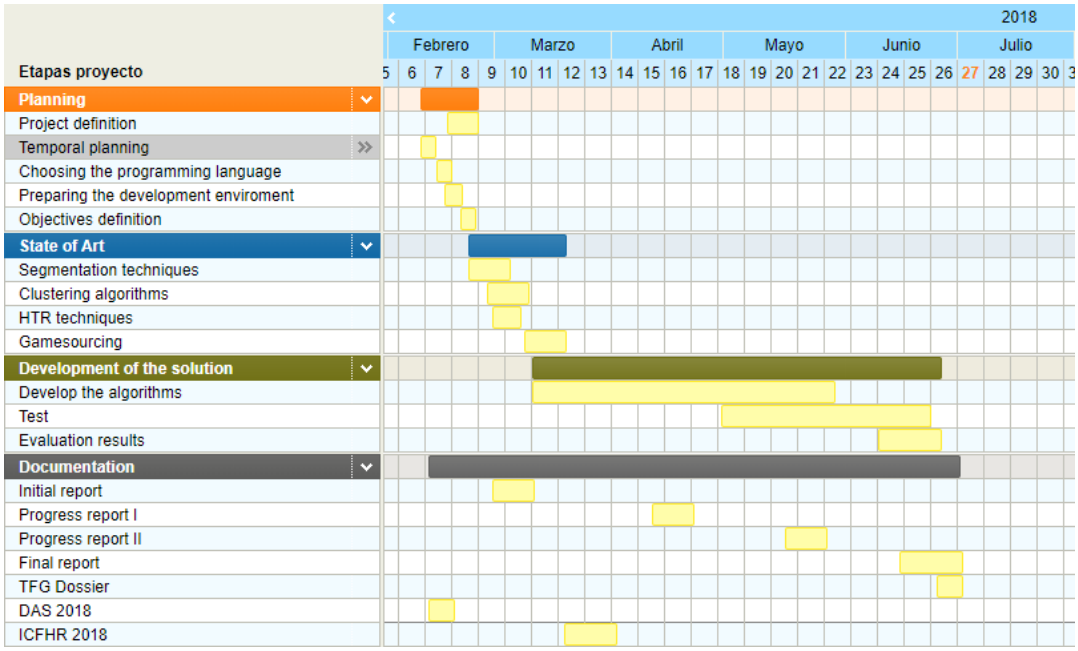

Fig. 18: Gantt diagram of the work.