Ensuring an error-free transcription on a full engineering tags dataset through unsupervised post-OCR methods

Mathieu François, Véronique Eglin

Ensuring an error-free transcription on a full engineering tags dataset through unsupervised post-OCR methods

Mathieu Francois[1,2] and Véronique Eglin[1]

[1] Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France
veronique.eglin@insa-lyon.fr
[2] Orinox, Vaulx-en-Velin, France
francois.mathieu@orinox.com

Abstract. The digital transformation of engineering documents is an ambitious research topic in the industrial world. The representation of component identifiers (tags), which are textual entities without a language model is one of the major challenges. Most of OCR use dictionary-based correction methods so they fail at recognizing hybrid entities composed by numerical and textual characters. This study aims to adapt OCR results on language-free strings with a specific semantics and requiring an efficient post-OCR correction with unsupervised approaches. We propose a two-step methodology to face the questions of post-OCR correction in engineering documents. The first step focuses on the alignment of OCR transcriptions producing a single prediction refined from all OCR predictions. The second step presents a combined incremental clustering & correction approach achieving a continuous correction of tags' transcriptions relatively to their assigned cluster. For both steps, the dataset was produced from a set of 1,600 real technical documents and made available to the research community. When compared to the best state-of-art OCR, the post-OCR approach produced a gain of 9 % of WER.

Keywords: Post-OCR correction · Incremental Clustering · Merging OCR · engineering P&ID diagrams

1 Introduction

In the industrial world, the creation of a digital twin of technical engineering documents is still a subject of interest. Research on this subject has been conducted for decades, but due to the complexity and lack of structure of these documents, the technical challenges in this area are still being studied by researchers around the world. Among these, precise transcription of texts without a language model, referred to as tags, is particularly sought after by industrial actors. Among these, precise transcription of texts without a language model, referred to as tags, is particularly sought after by industrial actors. These actors

need an error-free transcription of these texts in order to identify which component the tag represents and to refer to the project documentation for all the technical characteristics.

In this paper, we will attempt to provide innovative solutions to this problem in the form of an end-to-end system. First, by publishing a dataset representing this use case. Then, the proposed pipeline is divided into three main steps. The first one is a novel approach to align multiple OCR predictions. Next, we will propose an initial approach to tag correction based on clustering and correction through the tag structure. Finally, we will propose an evolutionary approach to our Post-OCR correction.

This paper is presented as follows: first, section two will present the related works on the topics of context retrieval on semantic-free texts. Section three will present the dataset submitted to the scientific community. The fourth section will present the different approaches proposed to reduce the number of transcription errors on semantic-free strings. And finally, part five will present the experiments performed with the results step by step. The results of this part are encouraging. Between the raw output of the OCR and the end of our end-to-end pipeline, we obtain a 9.16% gain in WER on the tags.

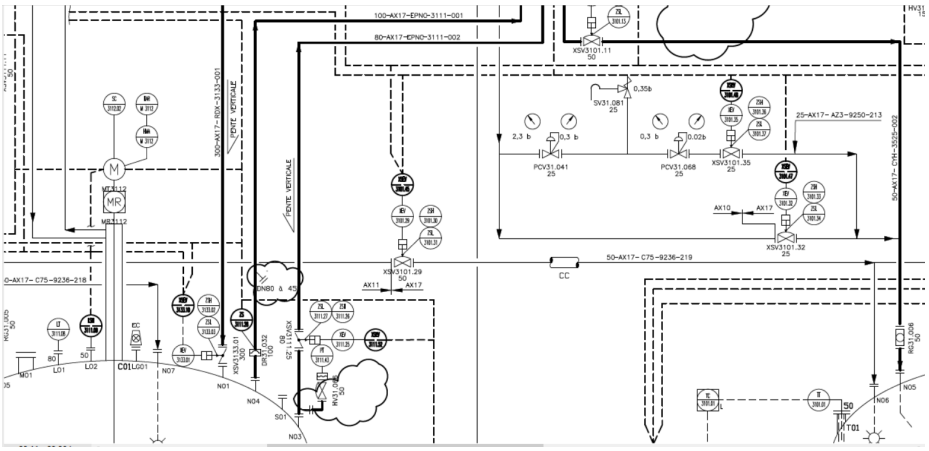

Fig. 1. Sample of a technical engineering drawing.

2 Related Works

Nowadays, OCRs have become very efficient, especially on printed text. OCR prediction is now considered a mature field and related work is now focused on post-OCR correction or more directly on the post OCR processing of even noisy outputs. If we consider the particular nature of the texts of the industrial *P&ID*

(Piping and Instrumentation Diagrams), beyond even the many noises that spoil the images of these texts, one must resolve to correct all OCR errors, even sparse, because they strongly compromise the quality of the indexation. In our industrial context, an accuracy close to 100% of correct recognition is explicitly required to assure secured industrial processes. A post-OCR processing is therefore definitely required.

2.1 Post-OCR processing and specificity of short-text

In recent years, the literature has presented many works focused on the impact of noisy transcriptions on information extraction or retrieval and NLP-based tasks (e.g. question answering, text summarization). In the general case of information extraction, it is essential to have transcriptions that are as faithful as possible to the original data, whether they correspond to long texts (complete semantic sentences) or short words (named entities, tags embedded in graphical documents, information present in tables, etc.). A notably valuable analysis was produced in 2021 by Van Strien et al. [14] confirms that a low quality of transcriptions of short texts had a negative impact on an information extraction/retrieval task. While information retrieval performance can remain satisfactory even with high WER (word error rate) estimates on long texts, it has been shown that the performance drastically decreases on short texts,[15].They attempt to prove that an error rate of 5% leads to linearly increase indexing (and thus information retrieval & extraction) errors. The main limitations that prevent OCR tools from reaching 100% accuracy mostly concern text background appearence (incrusted in colored backgrounds, presenting sometimes colorful patterns) ; blurry texts ; skewed or non-oriented documents ; presence of a large variety of letters (uncommon font types and sozes, rare alphabets, cursivity or handwritten-like aspects...) ; look-alike characters (OCR tools fail to distinguish between the number “0” and the letter “O” for example), [4]. This is a common criterion usually solved by the use of dictionaries or language models. A less common but highly significant factor can be added to this list. It is related to technical material (text-graphic documents, part lists, bills...) presenting texts written as fragments (partial words, hybrid sequences of letters and numbers...) with no apparent semantics from a linguistic point of view, but with a ” domain specific ” semantics that only an expert can decipher. Our work is part of this context. More precisely, it can be depicted as ”*isolated-word approaches*”, according to the taxonomy of Nguyen in [4]. For this class of approaches, the post-OCR correction techniques rely on observations of single tokens. Given the poor semantic and linguistic context of these tokens, merging OCR outputs techniques or lexicon-based approaches are often proposed,[18]. In our situation, the lack of contextual information around words (isolated tags) encourages to privilege competitive methodologies aiming at a selection of most suitable transcriptions. In that context, we can mention Lund et al.’s work in [1]. Authors proposed to use the A* algorithm to align different OCR predictions. Their algorithm can be used for shortest path problems but it can also be applied to text comparison. One of the strong points of this algorithm is that it will determine which solution is the most optimal without

having to explore all possible paths. Wemhoener et al. [2] also proposed their own alignment method, based on selecting a pivot from the OCR outputs. The other predictions are then individually aligned with the pivot. This makes it possible to link all predictions and to perform a comprehensive alignment. Broadly speaking, voting strategies or ensemble methods for multiple input selection have become well-established standard options for post-OCR error processing [16]. Nevertheless, as some small texts can be devoid of semantics (such as isolated tags in engineering diagrams), a voting strategy can also be combined with a lexical alignment technique [17]. Since our tag dataset could not be supplied with any lexicon and a voting strategy could not provide a reliable answer, we designed a hybrid technique combining a subsequence-based alignment of OCR outputs and the support of incremental clustering before correction.

2.2 Incremental similarity clustering

Since the documents in our study contain texts that are heavily lacking context - and thus do not allow for reliable transcriptions -, we turned to clustering approaches to identify frequent patterns/tags and contribute to their accurate recognition. The incrementality associated with clustering enables us to address real engineering situations where the data must be processed in a continuous flow. At the start of the process, we do not have enough training or initialization data that could guarantee an immediate and stable clustering. Clustering methods are widely used in many areas of science today. However, when data is constantly being added or updated, these methods can be limited. Most algorithms process all data in one pass, and when the information is updated, the clustering is recalculated on the updated data, resulting in a large consumption of resources. To reduce the computational costs and to better consider the evolution of data as they come in, *incremental clustering* approaches have been introduced. Prasad et al. [7] proposed an incremental adaptation of the K-Means algorithm. The principle of their approach is to adapt the value of the seeds one by one. The center of the cluster is updated at each iteration by minimizing the Sum of Squared Error (SSE), overcoming in that way the tedious initialization problem of k-means method. Authors show that this method allows a better composition of clusters with a total SSE estimated over all the clusters to be decreased.

Sun et al. [8] proposed an *Incremental Affinity Propagation* method ($\mathcal{AP}$). To evaluate their approach, they performed their experiments on real world time series. Their method uses the traditional *Affinity Propagation algorithm* to process the first data. Once inserted, data are processed through two approaches. The first one is based on an association between $\mathcal{AP}$ and K-Medoids, exploiting $\mathcal{AP}$ to define the clusters and K-Medoids to evolve them. The second approach proposed is an association between $\mathcal{AP}$ and the Nearest Neighbor Assignment. This last approach allows that two similar data not only belong to the same cluster but also have the same status.

Chakraborty in [9] has worked on an incremental *DBSCAN* method. This approach will initially form the clusters by given radius and minimum number of points per cluster. When the data is modified, the clusters are also updated by

calculating the minimum mean distance between the existing cluster data and the newly inserted or updated data.

Taking into account the scalability of the incoming data of a digitization chain, and the nature of the clusters (number and quality) obtained respecting the diversity of the tags of technical *P&ID* documents, we opted for a cooperation between $\mathcal{AP}$ (for the initialization step without any a priori number of classes) and an iterative cluster adjustment through incremental K-means (see section 4.3).

3 Tags_PID : a new public dataset for *P&ID* documents

Data from engineering documents are really sensitive. To our knowledge, there was no public annotated tags dataset that was truly representative of what can be found in the industrial world for *P&ID* documents. This section presents *Tags_PID*, a new tags image dataset obtained from real P&ID documents sources and made available to the scientific community[3].

3.1 Context

Tags_PID dataset is composed of tags images. The tags are the identifiers of the engineering technical components which are language model free and incomprehensible if we have neither engineering knowledge nor information about the project. These data come from real industrial projects from different sectors of activity such as: water treatment, gas and pharmaceutical.

The goal of *Tags_PID* is to represent a real industrial use case, which has been imagined as follows: An industrial engineer digitizes the drawings of a project and is particularly interested in tags. He has no indication of the tagging rules employed for this project. Considering the quantity of data to process, he wants a fully automated digitization process. Several days later, he retrieves new documents and wants to extract the tags from them. This engineer needs a transcription as close as possible to the ground truth to correctly identify the technical component.

The tag images and ground truth were originally extracted on searchable PDFs. Thanks to the PDFMiner library, the coordinates and the transcription of each text could be extracted via the documents' metadata. After that, each tag sub-image are converted into an image format and associated with their transcription.

3.2 Content of *Tags_PID*

Tags_PID is composed by data separated in 2 groups. The first one, named *Tags_1*, represents the tags from 30 drawings, i.e. 1570 images. The second one, named *Tags_2*, corresponds to the tags of 7 other drawings from the same project, i.e. 125 images.

[3] https://github.com/mathieuF789/dataset_tags

The images are named under the form: 'eval_X_Y.png' with 'X', an iterator representing the drawing from which the image comes from and 'Y', an iterator representing the text. The data is stored in a *.csv* format with the name of the image, the ground-truth and the different OCR predictions.

Fig. 2. Examples of tag images from *Tags_PID*.

To evaluate the quality of OCR predictions compared to the ground-truth, the CER and WER metrics were used. Even if some tags can be considered as several words when calculating the metrics, we fixed the fact that a tag always represents a single string that can include spaces. Below are the formulas to calculate the CER and WER with S the number of substitutions, D of deletions, I of insertions and N of reference characters. The results of the OCRs on the dataset can be found in section 5.1.

$$CER = (S + D + I)/N \tag{1}$$

$$\text{WER} = \begin{cases} 0 & \text{if } CER = 0 \\ 100 & \text{otherwise} \end{cases}$$

The WER is the most significant metric in this use case because it is important that there are no errors in the tag transcriptions. Indeed, the tags allow to refer to the technical documentation and to identify which component is represented on the drawing. If there is any error then the identified component and its characteristics will not be correct.

4 Our approach

The proposed approach is an end-to-end system and it is divided into several steps. The main objective of this approach is to ensure an error-free transcription on a engineering tags dataset thanks to OCR Merging and Post-OCR correction methods. Additionally, it will provide a continuous evolution of the system each time new data is inserted. First, this allows to correct the new data via the knowledge already acquired previously. Additionally, new tags can allow for the creation, deletion, or modification of existing clusters. This will allow for the correction of tags that were already present in the clusters and considered as correct. An explanatory illustration of the complete system can be found in Figure 3.

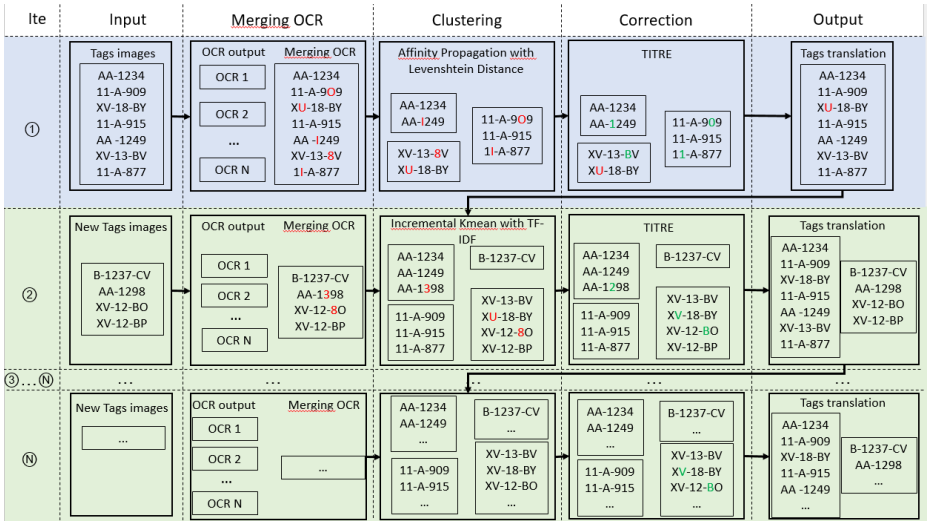

Fig. 3. Diagram of the proposed complete process.

4.1 Selection of the best OCR output

Our approach relies on the setting of multiple OCR views on a word input and the challenge is to select the best segments of each OCR output by focusing regions of mismatching. Given three OCR views of the same word i namely $S_1^i = (c_{1,1}^i, c_{2,1}^i, ...c_{n,1}^i)$, $S_2^i = (c_{1,2}^i, c_{2,2}^i, ...c_{n,2}^i)$, and $S_3^i = (c_{1,3}^i, c_{2,3}^i, ...c_{n,3}^i)$, where $c_{p,3}^i$ is the p^{th} characters of the third OCR prediction, our approach consists in two main steps:
(i) alignment of OCR outputs and spotting of differences and common substrings (determination of the common fixed basis of the predictions)
(ii) picking the majority choice for each substrings and concatenation into the final output.
Each sequence S_1^i, S_2^i and S_3^i is then decomposed into the same number of subsequences thanks to the *BaseFix* detection algorithm, thus leading to the division of S_1^i, S_2^i and S_3^i into $S_1^i = (id_{1,1}; s_{id_{1,1}}); (id_{2,1}; s_{id_{2,1}}); ...; (id_{m,1}; s_{id_{m,1}})$, $S_2^i = (id_{1,2}; s_{id_{1,2}}); (id_{2,2}; s_{id_{2,2}}); ...; (id_{m,2}; s_{id_{m,2}})$ and $S_3^i = (id_{1,3}; s_{id_{1,3}}); (id_{2,3}; s_{id_{2,3}})$; ...; $(id_{m,3}; s_{id_{m,3}})$, with for S_1^i m pairs $(id_i; s_i d_i)$ where $s_i d_i$ is a subsequence and id_i is a subsequence identifier in the word. In real situations, for large sequences composed by a maximum of 25 characters, we have $m <= 9$.

The goal is to segment predictions into consistent aligned substrings (common sub-sequences). We allow here small variations between two substrings from S_1^i, S_2^i and S_3^i transcriptions (addition, deletion, substitution of characters). To figure out the process, an example of sequences alignment considering three transcriptions for each word is presented in Figure 4. It is splitted into four steps also developed in the algorithm 1. The first step, *BaseFix Identification*, is the localisation of strictly identical substrings among the three predictions.

To be admitted as a fixed base, a substring must be composed of three or more characters and be present in all OCR outputs. These sequences are kept in memory and the remaining characters are isolated into secondary groups during the step *Sub-parts Isolation*. After this step, each group of strings is individually processed to determine the position of one or more potential offsets in the *Sub-parts alignment* phase. If the substrings are not the same size then the system will align them by adding a 'dummy' character. This one will be placed once again according to the similarities of the sequences calculated via a score system. Finally, we concatenate the aligned sub-sequences and the fixed bases previously kept in memory : the strings are now aligned. A voting system is finally applied to build the final string, which will serve as input of the next incremental clustering step.

Algorithm 1 Alignment of several OCR predictions for three OCR outputs

Require: Ω : list composed by S_1, S_2, S_3 the 3 OCR predictions
  B = findIdenticalSubSeq(S_1, S_2, S_3) /* baseFix identification */
  if B not $\varnothing$ **then**
    for $B_i \in$ B **do**
      for $S_j \in \Omega$ **do**
        $\Psi \cup \{ S_j \notin B_i \}$ /* sub-parts isolation */
      end for
    end for
  else
    $\Psi = \Omega$
  end if
  for $X_i \in \Psi$ **do**
    X_i = addCarac(X_i) /* sub-parts alignment */
  end for
  $\Omega = \{B \cup \Psi\}$ /* Concatenation */
  return Ω

4.2 Post-OCR Correction

In this section, a first phase of non-dynamic post-OCR correction is presented which is summarized in Algorithm 2. To propose a correction for OCR prediction without a language model, a first clustering of tag predictions had to be done. For this purpose, we used the same clustering algorithm as in our previous work based on Affinity Propagation, [5: anonymous reference]. To form the very first clusters in the first instance, we implement $\mathcal{AP}$ with the Levenshtein distance as metric (other could also been proposed but less adapted to string comparision. This metric proposes a very consistent clustering able to group tags of same composition even if they present punctual transcription errors.

Following this, we proposed a refinement of the clusters quality to improve the relevance of the corrections made to the tag. The first step was to define a

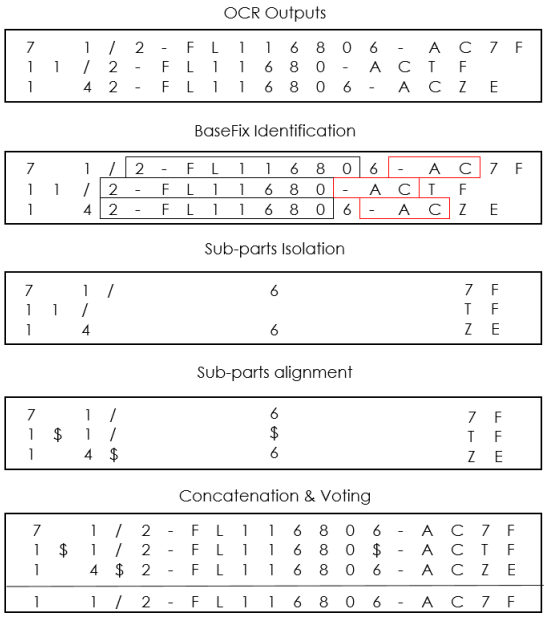

Fig. 4. Illustration of the the proposed alignment method in four steps.

regular expression that best represents each cluster. Tags that do not conform to the regular expression of the cluster from which they originate are placed in a rejection class. This last group of tags is isolated and two tag cleaning steps are performed in their respective clusters.
First, the process starts with the exploration of tags of each cluster and determines the similarities that we will call : *BaseFix*. In some cases, there is a hesitation if some characters are a *BaseFix* or not due to some tags not conforming to the whole cluster. We then use the *Ocr weighted Levenshtein distance* [10] to quantify the probability that this difference is due to a transcription error made by the OCR. From a threshold, we consider the error is real and we correct the corresponding tag(s). The second step is the identification of clusters where a recurring error in the OCR transcripts has occurred. This creates clusters with all tags having the same error. We study the proximity of all clusters in order to detect this case. If two clusters are very close and their difference, calculated using the *Ocr weighted Levenshtein distance*, is below a defined threshold, then the largest cluster is considered to be the one with the correct transcription. The tags considered false are modified and re-injected in the right cluster.

Following the refinement of the clusters, the tags of the rejection class can be analyzed. Using the alignment method proposed in section 4.1, we compare the tags with the regular expressions. We check for each tag if it can match the properties of a cluster with an addition, a deletion or a replacement of a

character. If it is the case, we make the choice to modify this character or not thanks to a score system based on more characteristics such as: is the character part of a BaseFix, the distance between the possibly false tag and the one it is closest in the cluster and finally, the consistency of the character to be added, deleted or replaced.

Algorithm 2 Post-OCR correction for tags

Require: T : list composed of tag transcriptions
  Γ = AffinityPropagation(T)
  E = $\varnothing$ /* regular expression representing the clusters*/
  K = $\varnothing$ /* reject class */
  for $C_i \in \Gamma$ **do**
    E = E $\cup$ { regEx(C_i) }
    for $s_j \in C_i$ **do**
      if $S_j \notin$ r **then**
        K = K $\cup$ S_j
        C_i = C_i - { $C_i \cap S_j$}
      end if
    end for
  end for
  Γ = transformWrongCluster(Γ)
  for $k_j \in$ K **do**
    if alignment(k_j, C_i) $\in C_i$ **then**
      if score(k_j, C_i) $\leq$ threshold **then**
        k_j = correc(k_j, C_i)
        C_i = C_i $\cup$ { k_j }
        K = K - K $\cap$ k_j
      end if
    end if
  end for
  T = $\Gamma \cup$ K
  return T

4.3 Contextualizing tags through incremental clustering

In accordance with the use case proposed in section 3.1, we wanted the tag clusters to be able to evolve if new data is injected. This has two main objectives: to continuously correct the tags by refining the delimitations of the clusters and also, to avoid starting the process again from scratch when new data are injected downstream from the first analysis. The incremental clustering track has therefore been studied.

An Incremental K-Means is used for the incremental part with the *term frequency-inverse document frequency* (TF-IDF) chosen as tags proximity measure. TF-IDF allows to evaluate the importance of a term contained in a set of text.

This second metric is complementary to the Levenshtein applied in the first clustering by $\mathcal{AP}$. By changing the metrics concerning the clustering methods, we will show in the experimental parts in what extend it brings a kind of freshness in the construction of clusters, especially it allows to detect mistranscribed tags that were not picked up in the first step.

Once the clusters were built, the same correction method as presented in part 4.2 when correcting the tags from the rejection class was applied. But depending on the constitution of the clusters, the results could be very different. This is the reason why all measurements are systematically taken on 40 batches.

5 Experiments and Discussions

In this section, we present the step-by-step results of the methods presented in Part 4. For each tag images, 4 OCR predictions are performed (EasyOCR[13], Tesseract[11], Paddle_OCR [12] and Paddle_OCR with binarized images). The results of these systems can be found in Table 1 on three datasets, *Tags_PID*, SROIE [3] and CORD [6].

The CER and WER metrics are used in this study to quantify the results. As mentioned before, we considered that a tag is a single string even if it contains spaces. Therefore, we have adjusted the WER calculation so that even the slightest error in the tag will set the WER to 100. This first table is not discussed here, as it will serve as a baseline for comparison with the changes brought about by our proposal.

Table 1. Comparing OCR results with different datasets.

Dataset	Tags_1		SROIE [3]		CORD [6]	
Method	CER	WER	CER	WER	CER	WER
EasyOCR (1)	8.71	53.79	14.06	43.19	17.09	38.74
Tesseract (2)	2.63	18.60	15.34	32.08	26.08	37.54
PaddleOCR (3)	**1.98**	**15.68**	**5.70**	**21.61**	**5.72**	**15.08**
PaddleOCR_b (4)	3.85	29.89	10.44	25.32	13.37	32.32

5.1 Merging OCR Results

In this section, we present and compare the results of the approach proposed in Section 4.1 (selection of the best OCR output). The OCR predictions on the different datasets (table 1) are used to the alignment phase of strings. They allow to form four different combinations consisting of three OCR predictions. For each of the combinations, the following alignment methods were applied: MinDist [1], Pivot [2] and our approach.

For each case, we apply the same voting system after the alignments so that the evaluation is as fair as possible. The voting method is basic (and works as

described in Section 4.1 by sliding slices of three characters, allowing for some shifts depending on the recognition results). We first sort the aligned strings according to the reliability of the OCR transcription. In other words, the first string will be the one from the OCR with the best results and the third will be from the OCR with the worst results, then, we apply the following condition : For a given word i, if a character $c^i_{p^j,2}$ (at the p^{jth} position) of the second string is equal to the character $c^i_{p^j,3}$ (at the p^{jth} position, with a possible negative or positive shift of 1) of the third string then the character of the final string will be this one, otherwise it will be the character $c^i_{p^j,1}$ of the first string (more confident result).

The results are presented in Table 2. The combination of OCRs is expressed as a number to lighten the table with : EasyOCR (1), Tesseract (2), PaddleOCR (3) and PaddleOCR_b (4). Our alignment method outperforms the 2 methods seen in the state-of-art. It is obvious to notice that the better the quality of the original OCR transcripts, the better the merging-OCR result.

Table 2. Comparing alignment methods using different OCR combinations.

Dataset	Tags_1						SROIE [3]						CORD [6]					
Method	MinDist		Pivot		Approach		MinDist		Pivot		Approach		MinDist		Pivot		Approach	
OCRs	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER	CER	WER
(1,2,3)	1.70	14.34	1.95	15.78	**1.37**	**13.73**	5.57	19.08	7.92	23.89	**5.15**	**18.69**	5.77	16.31	9.63	22.79	**5.48**	**16.17**
(1,2,4)	2.20	16.21	2.47	19.62	**2.00**	**15.49**	8.66	23.54	9.42	26.02	**8.41**	**23.66**	12.02	26.46	13.00	29.34	**11.33**	**26.16**
(1,3,4)	1.97	15.72	2.18	19.08	**1.63**	**15.44**	5.12	17.53	6.82	20.59	**4.71**	**16.72**	5.32	16.28	8.11	20.88	**4.92**	**15.46**
(2,3,4)	1.72	14.61	1.68	14.94	**1.39**	**13.78**	4.81	15.75	6.21	17.69	**4.37**	**14.55**	5.65	15.16	8.52	20.17	**5.16**	**14.26**

5.2 Post-OCR Result - First correction

As seen in the state-of-art, post-OCR correction of strings with poor semantics is a topic that is usually not addressed very well, as it concerns very specific situations of decontextualized data. Indeed, it is very hard to find methods to compare with that are not based on supervised learning, a lexical approach, or a language model. We consider here a very unique use-case but nevertheless essential in the engineering world.

Consequently, the overall comparison of our approach with public datasets is tenous to achieve. On datasets like SROIE [3], CORD [6], the data have no context between them, each image comes from a different environment. So, for this part, we only compare our proposition with the basic solution (based on AP Clustering) without incrementality.

The results presented in Table 3 have been realized on the Tags_1 dataset with the Merging OCR output (2,3,4) of Table 2 as input. We see a clear improvement quantified by a 6.5% gain on the WER.

Table 3. Results of post-OCR methods.

Methodes	CER	WER
Best result of 5.1	1.39	13.78
[5]	1.39	13.72
Approach	0.91	7.20

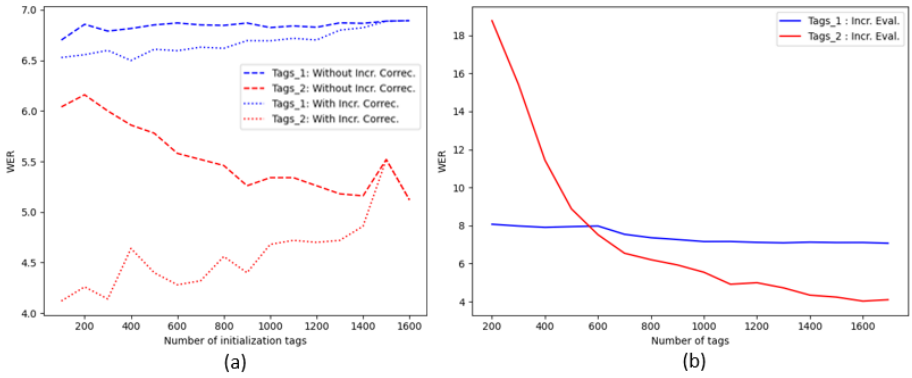

Fig. 5. (a) Comparing incremental correction and final correction. (b) Step-by-step evaluation of data insertion.

5.3 Incremental Clustering Results

To carry out these studies we used the 2 datasets presented in part 3. The blue curves represent the results for the dataset Tags_1. These are the tags treated in the previous parts and we take back the transcript of the tags from the output of the part 5.3. The Tags_2 dataset is represented by the red curve. The transcriptions of the tags of this dataset are done thanks to the OCR Merging seen in part 5.2. The results are available in Table 4.

The first study is the comparison between a correction performed when all the tags are inserted in the clusters and an incremental correction. Figure 4a. aims to measure the WER for a number n of initial tags (varying along x-axis). The x-axis represents the number of tags initially given to the Incremental Clustering algorithm and for each case, 200 tags are added at each new iteration. That is to say that for number of initial tags equal to 500 for example, the Incremental K-means will go through the iterations: 500,700,900...1700. The tags are inserted in a random way. Therefore, the results on a single case can differ completely depending on the order in which the tags are placed, which is why the results presented on the curve are the result of the average of 40 different distributions. The difference between the two methods is that the curves with the dashes were calculated thanks to the correction made when all the tags were sorted, whereas the curves with the dotted lines were calculated thanks to a correction phase

that was done at each iteration. For the incremental correction case, the best results are when the number n of considered tags is at its lowest. This seems quite obvious since there are more correction phases. The curve without incremental correction has an inverse trajectory. Depending on the distribution, the initial tags may not be very representative of all the tags. So, when we iterate too much on this basis the clusters are not ideal and the correction is less efficient. The curves logically meet at 1500 because there is only one iteration and therefore only one correction for both cases.

Figure 4b. represents the curve with an incremental correction, however the calculation of the metrics was performed at each iteration. As in the previous study, the results presented here are based on an average of 40 random tag distributions. We can see that both curves are descending and confirm that the more tags are inserted, the more representative the clusters are and therefore the better the correction. We just see a bigger progression for the second dataset, this is due to the fact that these tags were not processed by the part 5.3 and therefore more errors in the transcription were still present.

5.4 End-to-End Results

Table 4 shows the step-by-step results of each method presented above. PaddleOCR, which is supposed to be the best OCR we selected, reacts badly with the second dataset of tags. This is due to presence of many tags that are composed of spaces and these are poorly detected by PaddleOCR.

We have a 9.15% WER gain between the best OCR results and the end of the whole process for the first dataset. However, we can notice that the biggest increase is in the Post-OCR part. And when we inject the second set of tags, we have a gain of 24.68% of the WER between the output of the OCR Merging and the output of the incremental clustering and correction. These promising results allow us to conclude that on a perfectly representative set of real P&ID data, considering the data in batches and adapting the correction according to the evolution of the clusters outlines in an incremental way is much more effective than attempting a post-correction by operating the whole diversity of the dataset in one go. The gradual evolution of the description of the tag clusters is an additional dimension that contributes to the quality of the correction.

Table 4. Overall results of the proposed process on the *Tags_PID* dataset.

Dataset	Tags_1		Tags_2	
Methodes	CER	WER	CER	WER
PaddleOCR	1.98	15.68	25.15	79.03
Merging-OCR	1.39	13.78	3.45	28.8
Post-OCR	0.91	7.20	–	–
Incr. Clustering	0.83	6.53	0.25	4.12

6 Conclusion

In this paper, we propose an end-to-end approach to ensure the quality of the transcription of semantic-free strings from technical engineering P&ID documents.

First, we propose in this study a complete dataset of tags namely *Tags_PID* representing real textual entities what can be found in an industrial context. The dataset has been made available to the scientific community. This type of engineering data is quite sensitive and, to our knowledge, there was no public dataset of text/tag images with these specific characteristics.

After a merging-OCR method that aligns three different OCR predictions of tags and shows improvement compared to state-of-the-art results, our last contribution is a post-OCR correction method based on tag sorting using the Affinity Propagation algorithm with the Levenshtein distance metric. A correction method based on the tag structure and a Levenshtein distance weighted for OCR errors was then applied.

Finally, as we wanted our approach to be scalable, we applied an incremental clustering to improve correction efficiency using TF-IDF metric as distance between OCR predictions. This change of metrics introduced after the initialization step of the clusters by affinity propagation adds a new dimension to the evolution of the clusters boundaries and their generation. This incremental approach has also proven to be particularly effective as a support for post-OCR correction.

For future work, the next step is to study the graphical context around the tags and the interpretation of diagrams, providing additional information on the tags and helping to correct any remaining transcription errors.

References

1. W. B. Lund and E. K. Ringger. Improving optical character recognition through efficient multiple system alignment. In Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries (JCDL '09). Association for Computing Machinery, pp 231–240. (2009).
2. D. Wemhoener, I. Z. Yalniz and R. Manmatha, "Creating an Improved Version Using Noisy OCR from Multiple Editions" 2013 12th International Conference on Document Analysis and Recognition, pp. 160-164. (2013).
3. Z. Huang et al., "ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction," 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1516-1520. (2019).
4. T. T. H. Nguyen, A. Jatowt, M. Coustaty, and A. Doucet. Survey of Post-OCR Processing Approaches. ACM Comput. Surv. 54, 6, Article 124 (July 2021).
5. M. Francois, V. Eglin and M. Biou. Text Detection and Post-OCR Correction in Engineering Documents. In Document Analysis Systems: 15th IAPR International Workshop, DAS 2022, La Rochelle, France, May 22–25, 2022.
6. Park, S., Shin, S., Lee, B., Lee, J., Surh, J., Seo, M., Lee, H.: Cord: A consolidated receipt dataset for post-ocr parsing. In: Workshop on Document Intelligence at NeurIPS 2019 (2019).

7. Prasad, R., Sarmah, R. and Chakraborty, S. Incremental k-Means Method. (2019).
8. S. Leilei and G. Chonghui. Incremental Affinity Propagation Clustering Based on Message Passing. Knowledge and Data Engineering, IEEE Transactions on. 26. 2731-2744. (2014).
9. Chakraborty, Sanjay. Analysis and Study of Incremental DBSCAN Clustering Algorithm. International Journal of Enterprise Computing and Business Systems. 1. (2011).
10. Ocr weighted Levenshtein distance.Joan Capell Garcia. Version 1.0.0. July. 13, 2022. URL : https://github.com/zas97/ocr_weighted_levenshtein
11. R. Smith, "An Overview of the Tesseract OCR Engine," Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), pp. 629-633, (2007).
12. D. Yuning, L. Chenxia, G. Ruoyu, Y. Xiaoting, L. Weiwei, Z. Jun, B. Yifan, Y. Zilin, Y. Yehua, D. Qingqing and W. Haoshuang. PP-OCR: A Practical Ultra Lightweight OCR System. (2020).
13. EasyOCR. JaidedAI. Nov. 15, 2022. URL : https://github.com/JaidedAI/EasyOCR
14. D. van Strien, K. Beelen, M. Coll Ardanuy, K. Hosseini, B. McGillivray, and G. Colav izza. . Assessing the impact of OCR quality on downstream NLP Tasks. In Proceedings of the 12th International. (2020).
15. G. Torresan Bazzo, G. Acauan Lorentz, D. Suarez Vargas and V. P. Moreira. Assessing the Impact of OCR Errors in Information Retrieval. ECIR 2020: Advances in Information Retrieval pp 102–109, (2020).
16. S. Xu and D. Smith. Retrieving and combining repeated passages to improve OCR. In Proceedings of the 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL'17). (2017).
17. H. Gupta, L. Del Corro, S. Broscheit, J. Hoffart, and E. Brenner. Unsupervised Multi-View Post-OCR Error Correction With Language Models. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 8647–8652. (2021).
18. S. Rijhwani, D. Rosenblum, G. Neubig. Lexically Aware Semi-Supervised Learning for OCR Post-Correction. Computer Science, Transactions of the Association for Computational Linguistics. (2021).