Toward an Incremental Classification Process of Document Stream Using a Cascade of Systems

Joris Voerman[1,2(✉)], Ibrahim Souleiman Mahamoud[2], Aurélie Joseph[2], Mickael Coustaty[1], Vincent Poulain d'Andecy[2], and Jean-Marc Ogier[1]

[1] La Rochelle Université, L3i, Avenue Michel Crépeau, 17042 La Rochelle, France
{joris.voerman,mickael.coustaty,jean-marc.ogier}@univ-lr.fr
[2] Yooz, 1 Rue Fleming, 17000 La Rochelle, France
{joris.voerman,ibrahim.mahamoud,aurelie.joseph,
vincent.dandecy}@getyooz.com

Abstract. In the context of imbalanced classification, deep neural networks suffer from the lack of samples provided by low represented classes. They can't train enough their weights with a statistically reliable set. All solutions in the state of the art that could offer better performance for those classes, sacrifice in return a huge part of their precision on bigger classes. In this paper, we propose a solution to this problem by introducing a system cascade concept that could integrate deep neural network. This system is designed to keep as mush as possible the original network performance while it reinforces the classification of the minor classes by the addition of stages with more specialised systems. This cascade offers the possibility to integrate few-shot learning or incremental architecture following the deep neural network without major restrictions on system internal architecture. Our method keeps intact or slightly improves the performances of a deep neural network (used as first stage) in conventional cases and improves performances in strongly imbalanced cases by around +8% accuracy.

Keywords: Documents classification · Imbalanced classification · Deep learning · Image processing · NLP

1 Introduction

Private companies and public administrations have to manage a huge quantity of documents every day. These documents come from internal processes and from exchanges with external entities like subcontractors or the public administration. Processing so many documents requires a lot of human resources without an automatic system. In addition, these documents are generally linked to the administrative part or to the company's core activities. Such documents are consequently of primary importance as they generally validate an action or a decision inside and/or outside the company. The management of these documents

E. H. Barney Smith and U. Pal (Eds.): ICDAR 2021 Workshops, LNCS 12917, pp. 240–254, 2021.
https://doi.org/10.1007/978-3-030-86159-9_16

becomes a challenge between speed and precision. Indeed, an error could have a heavy cost by causing a wrong action or decision. Consequently, any automatic system that could be used in this context needs to have a high precision.

Many companies, like ABBYY, KOFAX, PARASCRIPT, or YOOZ, propose some document classification solutions known as a Digital Mailroom [5,25]. Those systems relies on a combination of deep learning methods and expert systems, where experts are needed to make those systems operational. Expert systems offer high performance in almost every situation, but with a very high maintenance cost to keep them up-to-date. They are then gradually replaced by machine learning methods where the performance is linked to the availability of large labeled datasets. These machine learning methods haven't entirely replaced expert systems, because in imbalanced cases the most reliable machine learning methods like deep learning lose a significant part of their performance and can't face to the lack of samples available to train them [30]. In the state of the art, this problem can only be solved by retraining the model each time enough samples have been gathered to properly train a new class.

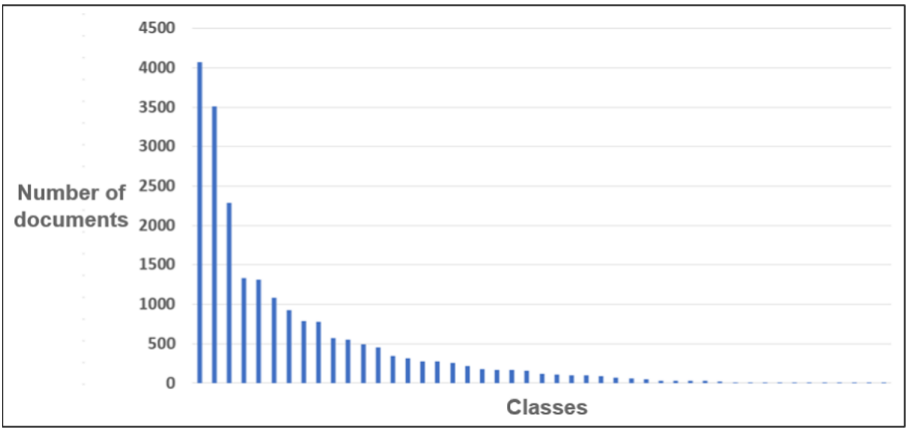

Fig. 1. Distribution of samples among all classes in our private database (a real document stream)

In order to formalize the document workflow, we use a document stream model proposed in [5]. A document stream is a sequence of very heterogeneous documents that appear irregularly over time. The stream is composed of numerous classes unequally distributed inside the stream with a group of core classes that is the majority of the stream and a lot of minor classes less represented. Figure 1 illustrates the distribution of documents among classes in a real document stream. Many of the minor classes have only few documents that do not allow a good training. Even if the number of documents per class increases as the content of a document stream evolves, new classes spontaneously appear

time-to-time and others disappear. So a training set generated from a document stream has two main properties: it is incomplete and imbalanced. In other words, it does not contain a sample for each class and the distribution of samples between classes is not equal.

The classification of documents issuing from a document stream is already studied in the literature, and the most recent and efficient approaches rely on machine learning and deep learning methods. Those methods have to deal with two main issues: the first one is the imbalance distribution of documents between classes. This leads to classes with a very few samples (sometimes only 1) to train the network while a few classes gather more than 70% of the total stream. In addition, the network has to be trained again each time a new class appears in the stream (which is far from being simple to detect). The second issue is precisely this addition of new classes, or samples with a really different template in an existing class. This second point has also been studied and some technology using incremental learning [3] or few-shot machine learning methods [31] have been proposed. However, even if they are able to learn new classes, the global performances on major classes (*i.e.* with the largest number of documents) did not compete with the elder methods.

Finally, the last challenge related to our context is the document classification itself. Indeed, this task is not simple and many articles have been published recently to adapt methods from image and language processing to the document classification. In fact, documents have the particularity to have two main modalities whose the importance varies according to the context: the image and the text. Here, with a majority of administrative and companies documents the text is the most important modality. Recently, a multi-modal approach of deep learning [2] appears to combine these two modalities. In the next section, we will review the state of the art around this theme.

2 Related Work

In the literature the document classification is traditionally divided into two approaches: text and image processing. For each of these approaches, many highly accurate deep learning architectures have been developed during the last years. The majority of newest deep learning methods for text classification use two steps. The first step is the generation of a word embedding because words are not suitable input for a neural network. The embedding is used to change a word into a numerical vector that could represent as much as possible the contextual word meaning. Multiple methods like Word2Vec [20], Fasttext [12] or BERT [6] have been developed to generate word embedding by training them on huge textual dataset like Wikipedia articles. The second step is a deep neural network trained on the targeted dataset. Many methods use a recurrent architecture like biRNN (bidirectional recurrent) [26] or RCNN (recurrent convolutional) [16] to take in account the sequence information of the sentence. These methods use specific recurrent neuron architectures with GRU [4] and LSTM [10]. Not recurrent methods also work, like textual CNN [13,33]. CouldScan [21] is a good example of these methods apply to the industrial context.

The majority of image processing methods used for classification are deep convolutional pixel-based neural network (CNN) with multiple architecture: VGG [27], NasNet [34], InceptionResNet [29], DenseNet [11], etc. Initially, those methods are designed for image classification, but they offer suitable performance on document classification. There are also some methods designed specifically for document classification [9]. Recently, a multi-modal strategy emerges and seems to take advantage of previous approaches. The state of the art proposes multiple combinations of previous deep neural network for a multi-modal classification [2]. Architectures are mainly two networks with one for text and the second for image that combine their output.

The one-shot and few-shot learning strategies [31] are also relevant in our case. The few/one-shot learning is a challenge in image processing which is defined as train with only a few numbers of samples per classes. Firstly, there is Bayesian based approach methods [7,17]. Secondly, some methods try to adapt the neural network architecture to few-shot learning like Neural Turing Machine and other Memory Augmented Network [23], Siamese Network [14] and Prototypical Network [28]. The incremental training also competes for this task, but the multiple tries to adapt neural networks for incremental learning [15,22,24] do not show great results. Older methods like incremental SVM [18], K-means [1] and Growing Neural Gas (IGNG) [3] seem to be more reliable for now. The literature shows some good samples of industrial applications based on these methods like INTELLIX [25] or INDUS [5].

At first sight, zero-shot learning [32] seems to be an interesting option to solve the problem of new class classification. However, zero-shot learning strategies are mainly designed for image classification and use transfer learning from a textual description or all other prior knowledge. In our case, we have no prior knowledge about new classes so these methods are unsuitable.

3 The Cascade of Systems

3.1 Main Idea

In order to solve the issue of low represented classes classification, we propose to introduce a cascade of systems. The objective is to keep the precision of deep learning network for the main classes (*i.e.* which represent the most frequent documents of the stream), while using more specific architectures designed for the least represented classes.

Our cascade follows a divide and conquer strategy, where the decision proposed by a system will be taken into account if and only if its confidence is high and this decision is reliable from a global point of view. More formally, the first stages of the cascade will deal with the most frequent classes, while the lowest stages will rely on other systems trained as a specialist to reinforce the classification of outliers and small classes. The cascade aims at promoting a high precision of each cascade stage. This also implies the use of a rejection system with a high threshold to send all rejected elements to the next stage of the cascade. The more we advance in the cascade, the more the systems become

specialized in rejected cases. This process is then repeated as many times as we need or want.

This strategy offers some advantages. Firstly, we have the possibility to combine the network with a system that could balance a deep learning network weakness, like an incremental method. The next stage system could equally use another modality than the previous one. This seems to offer better performance when modalities are complementary.

3.2 Training Architecture

The new problem introduced by the cascade is the training of the next stage system. Keeping the original training set to train the stage $n+1$ will mainly duplicate the results without significant benefit. The $n+1$ system will reject the same elements as stage n. To train stage $n+1$ effectively, we need to remove from the new training set all elements easily classified by the stage n and keep all possible rejected cases. We define this new training set in Eq. 1, where:

- D_n is the training dataset of stage n with $D_n \subset \{d_1, d_2, ...d_i\}$
- Each sample d_i has a class c_j
- D_{n+1} is the training dataset used by stage $n+1$ (*i.e.* that have been rejected at stage n)
- D'_n is all the documents with a very high confidence level and with a trustworthy class representation, according to stage n (will be detailed in the next section)

$$D_n = D'_n \cup D_{n+1} \tag{1}$$

The evolution of the training set from one stage to the next one acts as a focusing step for the next stage and imposes the system to better discriminate the documents rejected in the previous stages. In our architecture, each training phase is done successively as illustrated in Fig. 2. We operate this set division between each system training phase and then update the training set for stage $n+1$.

3.3 Set Division

With this architecture, the global performance of the next stages will depend on the division of the training set that becomes one of the main parts of the cascade. D_{n+1} needs to model as much as possible all the elements that will be rejected by the previous stage. This includes complex classes with a high intra-class variance, which overlaps with the closest classes (in the feature representation space), and that have not enough samples to train properly the model. This also includes outliers that have a lower confidence rate than other samples from the same class.

The selection process, which in practice corresponds to the confidence we grant to the system at stage n, is defined by a parametric selection function presented in Eq. 2. This selection function uses four different features to assess how

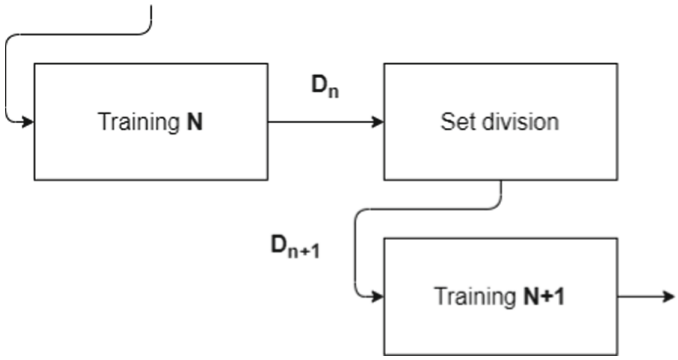

Fig. 2. Training architecture

the system performs at this stage on the proposed training set. These features gather information at the document level and at the class level.

We define D_{n+1} as $\forall d_i \in D_n$ with class c_j:

$$d_i \in D_{n+1}, \text{ if and only if, } \underbrace{\alpha A}_{\text{doc level}} + \underbrace{\beta B + \gamma C + \delta D}_{\text{class level}} < T \tag{2}$$

where:

- $\alpha, \beta, \gamma, \delta$ are weighting parameters, with $\alpha + \beta + \gamma + \delta = 1$.
- A is the confidence rate computed by the stage n system N_n for the corresponding class c_j. As deep neural networks generally propose high confidence rate, we use a normalization function to enlarge the confidence level.
  In accordance with the confidence result distribution given by the system we advise to use a min/max or an exponential normalisation. The objectives is to keep only the documents with the lowest score.
- B is the accuracy obtained by the stage n system N_n on class c_j. This feature is used to keep more documents from classes with a low accuracy.
- C is a ratio between the inter-class and the intra-class variance.
  Variances are computed with an euclidean distance between document's embeddings E_n trained by N_n and their class centroids. $Ed_n(d_i)$ is the distance between embedding of d_i and its class centroid, $Ed_n(c_j)$ is the same for class level.
  $\overline{Ed_n(c_j)}$ is the means of distances for class c_j, same for $\overline{Ed_n(D_n)}$ with the dataset. x_j is the number of samples within the class c_j and x_n number of class in D_n. As $C \in [0 : +\infty[$, we recommend to restrain values in $[0 : 2]$ then

normalize to keep all features in $[0:1]$.

$$\begin{cases} C = \dfrac{var_{inter}(c_j)}{var_{intra}(c_j)} \\ \forall c_y \in D_n \\ var_{inter}(c_j) = \dfrac{1}{x_n}\displaystyle\sum_{y=1}^{x_n}(Ed_n(c_y) - \overline{Ed_n(D_n)})^2 \\ \forall d_y \in c_j \\ var_{intra}(c_j) = \dfrac{1}{x_j}\displaystyle\sum_{y=1}^{x_j}(Ed_n(d_y) - \overline{Ed_n(c_j)})^2 \\ \overline{Ed_n(c_j)} = \dfrac{1}{x_j}\displaystyle\sum_{y=1}^{x_j} Ed_n(d_y) \end{cases} \tag{3}$$

– D is the representation of class c_j inside D_n with $size(c_j)$ the number of document in class c_j and $max(size(c))$ the number of document in the largest class. This feature is used to ensure that the lowest represented classes remain in D_{n+1}.

$$D = \frac{size(c_j)}{max(size(c))} \tag{4}$$

– T is a threshold parameter

The division of the training set is done once the stage n system training phase is completed. Consequently, the features are computed when the whole system is fully trained. The global proposed architecture is summarized in Fig. 3.

The selection function is applied to the training set and the validation set in order to ensure a fair validation process for the stage n system and to be sure that the training step of the stage $n+1$ system will focus on the remaining samples. The last step of our architecture is related to the reduction of the training and validation size. The balance between them will be broken in the majority of cases because the validation set become proportionally greater than the training set. We adjust them class by class by randomly choosing enough samples to adjust the balance between set. This need to be done class by class, because each class is not reduced with the same rate.

3.4 Decision Process

The decision process starts once the training phase is ended for all the stages of the cascade. The decision process uses a rejection system to separate the less reliable answer from the others. The rejection is applied on the confidence rate returned by the system (in the one-hot vector case, the confidence rate is the highest score of the vector). As mentioned earlier in this paper, our document stream classification problem entails a really high precision rate to avoid mistakes. To this end, we chose to set a high value for the threshold. All rejected

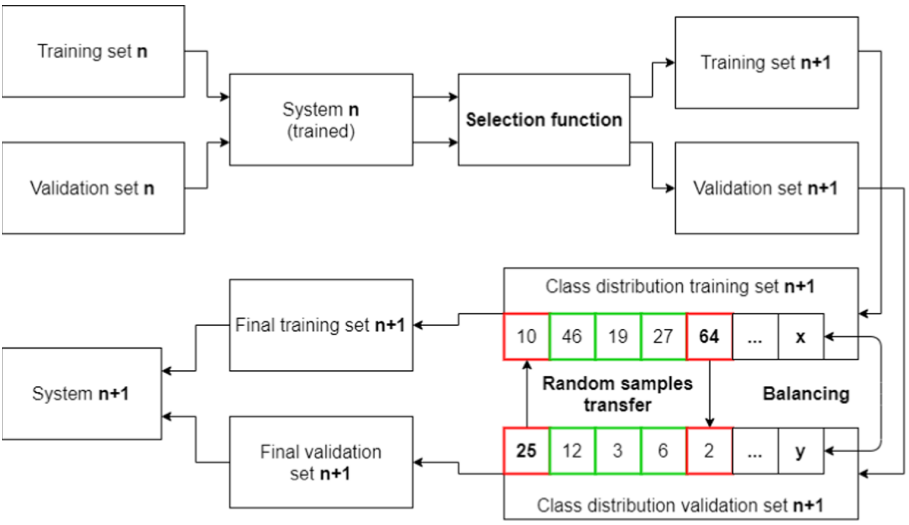

Fig. 3. Set division architecture

results are then sent to the next stage of the cascade that will apply the same process. If no decision is taken when the document goes through the last stage of the cascade, the document is finally labeled with a "reject label".

The last adjustment of our architecture concerns the computation of the final confidence rate in an imbalanced context. Indeed, the confidence rate returned for a class trained with two samples will not have the same value than the confidence rate returned for a class with five thousand samples. In addition, all cascade stages will potentially not be trained on all classes. To take this into account, we propose to weight this confidence rate. The main idea is to use the validation set to assess the reliability of our cascade stage for each class and adapt the confidence rate proportionally to the accuracy score of the class. This system will increase the rejection rate on classes that we estimate with the validation set to have the lowest accuracy and so the highest risk to give a wrong answer. The classes with the lowest accuracy are theoretically the same as those that were sent to the stage $n+1$ training set by the selection function. The new weighted confidence score of a document d_i (denoted as $N'_n(d_i)$) is computed by Eq. 5. It is composed of the former confidence score $N_n(d_i)$ and a penalty score. This penalty is based on the stage n accuracy $Acc_n(c_j)$ on validation set for the predicted class c_j. An additional r parameter tends to limit the class weight effect into specific bound. This parameter prevents the system to reject only more documents of low reliable classes and not whole classes.

$$N'_n(d_i) = \underbrace{N_n(d_i)}_{\text{Confidence score of } d_i} - \underbrace{r(1 - Acc_n(c_j))}_{\text{Weighted penalty Wb}} \tag{5}$$

The Fig. 4 illustrates the proposed decision process architecture with the new weighted scores system. The system will generally reduce a bit the stage n accuracy and increase the precision, mainly on low represented classes. The Next stages of the cascade will compensate the loss of accuracy as we will demonstrate in the next section.

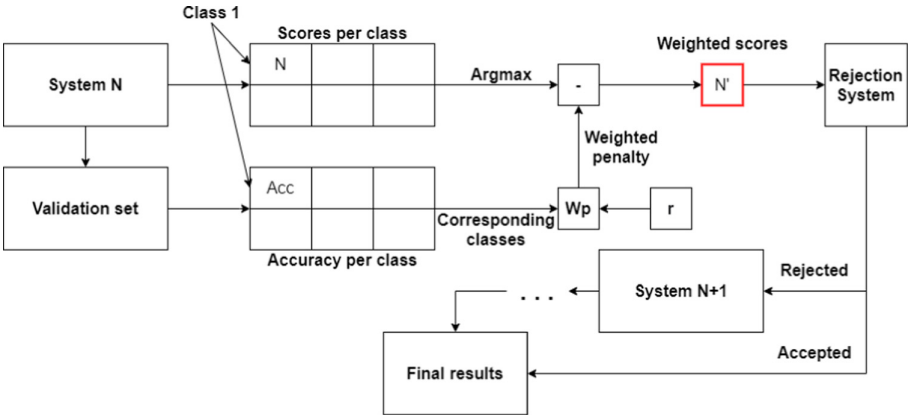

Fig. 4. Cascade decision with weighted rejection

4 Experiments

4.1 Architectures Used

All experiments done in this paper use a two stage cascade adapted for our context (private dataset). For the first stage, we used deep neural networks for their abilities to classify classes with a large number of labeled data. For the second stage, we propose more specific architectures able to deal with few samples per class during the learning phase. The selection has been done with one method per modality. To validate our ideas, we use two different networks in the first stage. A VGG16 [27] network as an image-based classification model and a textual-based bidirectional recurrent network (biRNN) [26] with CamemBERT [19] as embedding model. The biRNN is reinforced with an attention model to improve the trained features as proposed in [8]. These two methods correspond to state of the art deep neural network models for their respective modality classification task.

For the second stage, we propose to compare a few-shot learning method called prototypical network (ProtoNet [28]) known as one of the best few shot learning method, and an incremental learning approach (growing gas based method) called A2ING [3] and developed for document classification. The first method is based on visual features while the second one uses raw text from OCR as input. Those systems are selected to offer a maximum of variety in

modality and state of the art architectures designed for documents, image or text classification in the experimentation results.

Finally, and in order to allow a fair comparison of our proposed cascade architecture performance, we also used two other methods in our baseline. An image-based holistic CNN designed for document classification (HCNN [9]) and a non-recurrent textual network (called TCNN). The TCNN network is a combination of architecture from [13] and [33].

4.2 Datasets

To evaluate our system we used two datasets, one private and one public. The first is a private document database provided by Yooz company to assess the performance on a real-case document stream. We will call it the "YoozDB" set. It gathers real documents coming from Yooz's customers, with an access to scanned image of document and text extracted by ABBYY-FineReader (version 14.0.107.232). It is composed of 23 577 documents unequally distributed in 47 classes, with 15 491 documents (65.71%) for training, 2203 (9.34%) for validation and 5883 (24.95%) for test. Classes are imbalanced with between 1 to 4500 documents for each class (the documents distribution per class is proposed in Fig. 1). The classes includes multiple variation of invoice, business order, dues, tax notice, bank details, contract, mail, cheque, ID card, etc. The main drawback of this dataset comes out from this test set which is as the same distribution as its training set with some classes composed of only one (or very few) document. This leads to a lack of samples for the very low represented classes that do not offer statistically reliable results for those classes and reduce the effect on global performance.

The second dataset we used in our comparison is the well-known RVL-CDIP dataset [9]. Composed of 400 000 industrial documents gatherers from tobacco companies equally distributed in 16 classes: letter, memo, email, file-folder, form, handwritten, invoice, advertisement, budget, news, article, presentation, scientific publication, questionnaire, resume, scientific report and specification. Some of them have almost no text and overall this dataset is built as an image dataset and so disadvantages the text based methods. RVL is divided in a training set of 320 000 documents (20 000 per classes), a validation and a test set of 40 000 documents each (2500 per classes).

4.3 Experimentation Methodology

To evaluate the performance of our cascade architecture and all the methods used in our comparison, we use three classical information retrieval metrics: the accuracy, the precision and the rejection rate. We use accuracy to compare our results to the other methods of the state of the art. The precision is important for our context and finally, the rejection rate aims at displaying the relevance of our proposed rejection system.

All results on YoozDB are computed in a k-fold cross-validation process (in our case, k = 10 with seven folds for train, one for validation and two for test)

to deal as much as possible with low amounts of documents. Folds are randomly generated class by class to take into account the imbalanced distribution between classes.

In addition, we apply specific strategies to evaluate system adaptation to imbalanced and incomplete context of document stream. Those strategies follow the same methodology as that proposed in our previous work [30]. Briefly, we generate two new sets called Imbalanced and Realistic from RVL-CDIP. The first evaluates the resilience to imbalanced class distribution and so modifies the original set to simulate the imbalanced distribution of YoozDB. The modification is done by gathering all classes in four groups of four random classes and each group keeps only a part of their samples (5%, 10%, 50% and 100%). The second, add a group of classes with only one sample to represent classes that will appear later in the document stream. The distribution between groups becomes four classes with only one document and three for each of the imbalanced groups as previously. All these modifications are done only on the train and validation sets, but not on the test set to keep the results as reliable as possible.

4.4 Results

All experimental results on a balanced RVL-CDIP and YoozDB sets are presented in Table 1. We can observe that all the results given by cascade's methods are at least equivalent to basic methods. One exception appears on the biRNN-ProtoNet architecture on RVL-CDIP that offer better performances. This upgrade is linked to the fact that modalities complementing each other between the biRNN (text) and the ProtoNet (visual). The gain is mainly on file-folder and handwritten classes that are very difficult for a text classifier due to the lack of text. The results on YoozDB are not very significant here because, as explained previously, the test set is imbalanced and incomplete. In fact, the under-representation of minor classes in the test set limits or nullifies the impact of these classes on the global result. In addition, it is impossible to evaluate the real internal diversity with only two samples of the same document template and at least the results are not statistically reliable on these classes. Consequently, the results on this dataset come mainly from a good classification of major classes with a less importance of imbalanced, and global performance remains close to balanced dataset. The result of A2ING combines on RVL-CDIP are not displayed, because this method is not adapted to RVL documents that lack of words for a majority of cases. Finally, the multi-modal potential seems to be limited. The other methods from the state of the art have better gain for the combination of text and image modality in document classification task [2]. The results given by the table for VGG16 and biRNN alone can be used as raw performance of the first stage (n) of our cascades (respectively to their combination). If you want, it is possible to calculate the second stage ($n + 1$) only performances from combining result and corresponding raw method.

On the second table (Table 2) we have all results of methods computed on RVL-CDIP modified to simulate an imbalanced set with the protocol introduce in Sect. 4.3. For the imbalanced case, the cascade system offers a +11% accuracy

Table 1. Balanced case results

Dataset	RVL-CDIP			YoozDB		
Method/Measure	Accuracy	Precision	Reject rate	Accuracy	Precision	Reject rate
HCNN	**80.76%**	**95.42%**	15.36%	84.57%	95.67%	11.61%
TCNN	65.52%	93.72%	30.09%	88.36%	98.97%	10.72%
VGG16	75.14%	94.41%	20.41%	84.70%	95.47%	11.28%
VGG16 - ProtoNet	75.79%	94.04%	19.41%	84.21%	95.61%	11.92%
VGG16 - A2ING	–	–	–	82.05%	96.62%	15.08%
biRNN	77.95%	88.58%	12.01%	93.57%	99.22%	5.69%
biRNN - ProtoNet	80.72%	88.99%	**9.30%**	**93.77%**	98.95%	**5.23%**
biRNN - A2ING	–	–	–	91.45%	**99.30%**	7.90 %

for the VGG16 and +6.5% accuracy for biRNN against around −1% precision. It means that the cascade recovers a third of documents rejected by VGG16, and close to the half for biRNN, with almost the same precision in an imbalanced context. With absolute values, the cascade system recovers more than 5 000 documents with the same precision as the network alone. This means 5 000 documents less to process manually of 40 000. This gain comes from the better adaptation of 5-shot learning methods like ProtoNet to train classes with fewer samples and the rebalancing done by the cascade. Indeed, an important part of major class samples has not been transferred to the second stage training set, so minor classes become proportionally more represented. For example on VGG16, the cascade with ProtoNet recover 43.52%, 17.16%, 29.16% and 13% of rejected document for the classes that was reduced to 5% of training document as you can see with Table 3. The results for realistic case are less impressive, even if the accuracy earns +8%, because the precision is more reduced by the second stage. In fact, the Prototypical Network is less effective in the one-shot learning situation.

Table 2. Imbalanced case results

Dataset	Imbalanced			Realistic		
Method/Measure	Accuracy	Precision	Reject rate	Accuracy	Precision	Reject rate
HCNN	67.97%	89.14%	23.75%	55.70%	77.55%	28.18%
TCNN	51.98%	**91.63%**	43.27%	36.88%	**78.50%**	53.02%
VGG16	59.17%	88.90%	33.44%	51.28%	77.97%	34.24%
VGG16 - ProtoNet	70.19%	87.67%	19.94%	**58.89%**	73.28%	19.64%
biRNN	68.37%	79.73%	14.25%	50.71%	67.56%	24.94%
biRNN - ProtoNet	**74.90%**	79.00%	**5.19%**	58.05%	64.30%	**9.72%**

Table 3. Differences per classes between VGG and VGG-ProtoNet cascade on imbalanced RVL-CDIP

Groups	100%				50%			
Classes	Advert	File F	Handwr	Sc report	Budget	Email	Invoice	Resume
Precision	0.06%	−5.97%	−5.80%	0.69%	−1.45%	−0.25%	−0.06%	−0.23%
Reject rate	−4.60%	−7.00%	−2.12%	−5.00%	−6.52%	−1.72%	−3.20%	−3.44%
Groups	10%				5%			
Classes	Form	Letter	Presen	Questi	Memo	News A	Sc Public.	Speci
Precision	−8.16%	−1.98%	−5.48%	−2.02%	−1.27%	−4.05%	−3.51%	−1.10%
Reject rate	−18.40%	−17.76%	−22.88%	−20.56%	−13.00%	−29.16%	−43.52%	−17.16%

5 Conclusion

5.1 Overview

We propose a new architecture to combine successively in a cascade multiple neural networks to reinforce them in an imbalanced context. A two stages combination between a deep network and a 5-shot learning method offers between +11% and +8% accuracy in imbalanced context and keeps equivalent performance in balanced cases. This cascade method offers multiple possibilities of combination and do not limit too much the internal architecture of combined systems. It needs an embedding representation of the document and a class confidence score to compute the next stage training set. The results of the cascade system in imbalanced context are promising but need a better adaptation for one-shot cases.

5.2 Perspectives

In perspectives, we will try to combine our cascade with a multi-modal architecture and evaluate the potential of a cascade with more than two stages. Equally, we want to explore further the potential of a neural network cascade with an incremental system and build a better synergy between them to create an incremental deep learning system with high performances. In addition, we consider designing a methodology to assess the quality of the next stages training set and so the selection function. All parameters used for our experiments have been selected empirically for now. We will find a way to compute them automatically in future research.

References

1. Aaron, B., Tamir, D.E., Rishe, N.D., Kandel, A.: Dynamic incremental k-means clustering. In: 2014 International Conference on Computational Science and Computational Intelligence, vol. 1, pp. 308–313. IEEE (2014)
2. Bakkali, S., Ming, Z., Coustaty, M., Rusinol, M.: Visual and textual deep feature fusion for document image classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 562–563 (2020)

3. Bouguelia, M.R., Belaïd, Y., Belaïd, A.: A stream-based semi-supervised active learning approach for document classification. In: 2013 12th International Conference on Document Analysis and Recognition, pp. 611–615. IEEE (2013)
4. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
5. d'Andecy, V.P., Joseph, A., Ogier, J.M.: IndUS: incremental document understanding system focus on document classification. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 239–244. IEEE (2018)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. IEEE Trans. Pattern Anal. Mach. Intell. **28**(4), 594–611 (2006)
8. Górriz, M., Antony, J., McGuinness, K., Giró-i Nieto, X., O'Connor, N.E.: Assessing knee OA severity with CNN attention-based end-to-end architectures. arXiv preprint arXiv:1908.08856 (2019)
9. Harley, A.W., Ufkes, A., Derpanis, K.G.: Evaluation of deep convolutional nets for document image classification and retrieval. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 991–995. IEEE (2015)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
12. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651 (2016)
13. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
14. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML Deep Learning Workshop, Lille, vol. 2 (2015)
15. Kochurov, M., Garipov, T., Podoprikhin, D., Molchanov, D., Ashukha, A., Vetrov, D.: Bayesian incremental learning for deep neural networks. arXiv preprint arXiv:1802.07329 (2018)
16. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)
17. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science **350**(6266), 1332–1338 (2015)
18. Laskov, P., Gehl, C., Krüger, S., Müller, K.R.: Incremental support vector learning: analysis, implementation and applications. J. Mach. Learn. Res. **7**, 1909–1936 (2006)
19. Martin, L., et al.: CamemBERT: a tasty French language model. arXiv preprint arXiv:1911.03894 (2019)
20. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
21. Palm, R.B., Winther, O., Laws, F.: CloudScan-a configuration-free invoice analysis system using recurrent neural networks. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 406–413. IEEE (2017)
22. Rosenfeld, A., Tsotsos, J.K.: Incremental learning through deep adaptation. IEEE Trans. Pattern Anal. Machine Intell. **42**, 651–663 (2018)

23. Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T.: One-shot learning with memory-augmented neural networks. arXiv preprint arXiv:1605.06065 (2016)
24. Sarwar, S.S., Ankit, A., Roy, K.: Incremental learning in deep convolutional neural networks using partial network sharing. IEEE Access **8**, 4615–4628 (2019)
25. Schuster, D., et al.: Intellix-end-user trained information extraction for document archiving. In: 2013 12th International Conference on Document Analysis and Recognition, pp. 101–105. IEEE (2013)
26. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11), 2673–2681 (1997)
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
28. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. arXiv preprint arXiv:1703.05175 (2017)
29. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
30. Voerman, J., Joseph, A., Coustaty, M., Poulain d'Andecy, V., Ogier, J.-M.: Evaluation of neural network classification systems on document stream. In: Bai, X., Karatzas, D., Lopresti, D. (eds.) DAS 2020. LNCS, vol. 12116, pp. 262–276. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57058-3_19
31. Wang, Y., Yao, Q., Kwok, J.T., Ni, L.M.: Generalizing from a few examples: a survey on few-shot learning. ACM Comput. Surv. (CSUR) **53**(3), 1–34 (2020)
32. Xian, Y., Schiele, B., Akata, Z.: Zero-shot learning-the good, the bad and the ugly. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4582–4591 (2017)
33. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in Neural Information Processing Systems, pp. 649–657 (2015)
34. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8697–8710 (2018)