Use of Language Models for Document Stream Segmentation

Chems Eddine Neche, Yolande Belaïd and Abdel Belaïd
Vandoeuvre-Lès-Nancy, F-54506, France

Keywords: Word Embeddings, Stream Flow Segmentation, Pages Pair Classification, Continuity, Rupture.

Abstract: Page stream segmentation into single documents is a very common task which is practiced in companies and administrations when processing their incoming mail. It is not a straightforward task because the limits of the documents are not always obvious, and it is not always easy to find common features between the pages of the same document. In this paper, we seek to compare existing segmentation models and propose a new segmentation one based on GRUs (Gated Recurrent Unit) and an attention mechanism, named AGRU. This model uses the text content of the previous page and the current page to determine if both pages belong to the same document. So, due to its attention mechanism, this model is capable to recognize words that define the first page of a document. Training and evaluation are carried out on two datasets: Tobacco-800 and READ-Corpus. The former is a public dataset on which our model reaches an F1 score equal to 90%, and the later is private for which our model reaches an F1 score equal to 96%.

1 INTRODUCTION

Several agencies and administrations receive a mass of documents in the form of a flow of pages which they have to process in a relatively short time. This is the case of incoming mail processing, the management of bank loans to find the supporting documents, etc. Once scanned, this gives rise to a continuous stream of images that need to be re-grouped into documents, which is not an easy task. Indeed, the pages that follow each other and that belong to the same document do not always show common characteristics and it is necessary to deeply explore them to find the document limits. The Figure 1 resumes the document stream segmentation process.

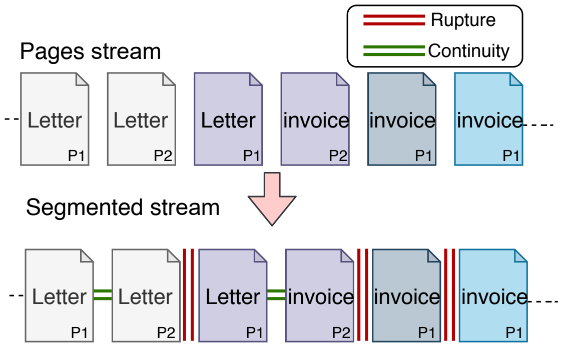


Figure 1: Documents stream segmentation task.

Existing works in the literature on documents stream segmentation can be divided into two main categories. The first one is based on the detection of ruptures (breaks) and continuities (Daher and Belaid, 2014; Karpinski and Belaid, 2016; Hamdi et al., 2018). In this case, the system is iterated on the successive pairs of pages, takes as input the current pair of pages, looks at whether the two pages contain indices of similarity or continuity, then, decides whether they represent a rupture or a continuity. The second one is based on page classification (Gallo et al., 2016), according to the assumption that for two successive pages belong to the same document, they must belong to the same class. After having all the classes of stream pages, consecutive pages that belong to the same class are grouped as a single document. The issue of this segmenting method is that the system is unable to properly separate two documents of the same class that follow each other in the stream. For example, two invoices will be grouped as one document which constitutes an error.

The two segmentation categories mentioned previously require an adequate system input modeling. Two proceeding ways are related in the literature. The first one consists in manually extracting document page descriptors, such as textual information (i.e. keywords), visual information (i.e. font style, paragraph or word location), logical (i.e. title section numbers, items, section continuation) and factual information (i.e. dates, introductory phrase, courtesy form in letters) (Daher and Belaid, 2014; Karpinski

Neche, C., Belaíd, Y. and Belaíd, A.
Use of Language Models for Document Stream Segmentation.
DOI: 10.5220/0009146402200227
In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2020)*, pages 220-227
ISBN: 978-989-758-397-1; ISSN: 2184-4313

and Belaid, 2016). Descriptors extraction process is tricky because it relies on set of regular expressions and rules. In addition, prior knowledge on processed document, is required. The second one, based on deep learning, tries to learn the descriptors directly. CNN (Convolutional Neural Network) has been used advantageously in this case because of its efficiency in visual feature learning in images. In contrast, for textual descriptors, the literature is full of techniques such as BoW (Bag Of Words), Word2Vec, Doc2Vec and word embedding (Wiki, 2017). These models are efficient for sentiment analysis and text classification.

In all our previous work, we used the classification of page pairs in continuity and breaks. We continue to do so because of its superiority over the page classification technique. But, we reinforce this technique in different ways, first by a correct representation of the content by using model languages as Doc2Vec, Word2Vec and word embedding. Then we use deep learning models like GRUs reinforced by an attention mechanism.

The rest of the paper is organized as follows. Section 2 describes the main techniques reported in the literature. The proposed approach is described in section 3 with the different models. In section 4, we will give a description of the used datasets, while in section 5, we will resume the experimental protocol and the obtained results. We will conclude and give future perspectives in section 6.

2 RELATED WORKS

The first works reported in the literature rather consider the flow as a sequence of pages, and the documents are found by sub-sequence analysis. Probabilistic models are used to model and recognize these sub-sequences.

So, in our research team, Meilender (Meilender and Belaid, 2009) used a method similar to the Variable Horizon Models (VHM) or multi-grams used in speech recognition. It consists in maximizing the probability of the flow using all the Markov models of the constituent elements (pages). Since the calculation of the probability of all pages is NP complete, the solution has led to the use of windows to reduce the number of observations. In Schmidtler and Amtrup (Schmidtler and Amtrup, 2017), single pages are characterized by bag of words. According to the authors, the discriminating features are located in the first and last page of a document. Therefore they model the document types by using three symbols: start, middle and end. Multi-class SVMs are used and their scores are mapped into probabilities. The probable best sequence of documents is extracted by using an algorithm similar to the beam search algorithm (Furcy and Koenig, 2005). Gordo et al. (Gordo et al., 2013) use an approach combining the multiple pages of a document into a single feature vector representing the document as a whole. Then, the most plausible segmentation of a page flow into a multi-page document sequence is achieved by optimizing a statistical model that represents the probability of each segmented document of several pages belonging to a particular class.

The second wave of work focuses on page pairs and tries to find out if they represent document boundaries. In (Daher and Belaid, 2014), a feature extraction process is used to construct the pair page descriptor which summarizes the pair page relation in term of rupture and continuity. This system classifies the pair descriptor into rupture or continuity using Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP). In the continuation of (Daher and Belaid, 2014), (Karpinski and Belaid, 2016) used a rule based system to detect ruptures and continuities in a hierarchy of documents from records (simple page), to technical documents, fundamental documents and cases (set of documents belonging to the same person). For each level, the system descriptors are first extracted and then compared between pairs of pages or documents. These descriptors can be section numbers, page numbers, dates, salutation and conclusion formulas. The technique in (Hamdi et al., 2017) and (Hamdi et al., 2018), uses Doc2Vec model to realize the segmentation task. At first, the Doc2Vec is trained to learn the documents pages representation. While sweeping through the stream pages, the system calculates the cosine distance between the page pairs. Finally the system compares this distance with a fixed threshold to determine if the pair represents a rupture or a continuity.

More recently, one can find in the state of the art deep learning techniques with convolutional neuronal models for the classification of documents, as (Gallo et al., 2016; Harley et al., 2015; Noce et al., 2016; Wiedemann and Heyer, 2017). While the first two use only textual information, the last two use textual and visual information. In (Wiedemann and Heyer, 2017), for the visual content, VGG16 is employed to learn document visual features. As for the textual content, a CNN of text data (Kim, 2014) is used. Then both results are combined to decide the segmentation type.

These last methods naturally led to reflect on the representation of documents to feed these types of models. The word embedding (mapping of words into numerical vector spaces) has proved to be an incredibly important method enabling various machine learning models that rely on vector representation as input data to enjoy richer representations of text input. These representations preserve more semantic and syntactic information on words, leading to improved performance in almost every model. This led researchers to consider the problem of how to provide richer vector representations to larger units of texts. This effort has resulted in a slew of new methods to produce these mappings, with various innovative solutions like Doc2Vec or Word2Vec.

3 PROPOSED APPROACH

Based on our previous discussion of the richness of textual content and the interest of having a good representation of this content, we have conducted several experiments incorporating textual page representations, that we will describe in the following subsections.

3.1 Doc2Vec + LSTM

Doc2Vec model has played a main role in text classification and sentiment analyses. This is why we are exploiting this document representation in order to create a stream segmentation model. This model uses a pre-trained Doc2Vec to calculate a page representation vector for the page pair (i.e. precedent and current page). The precedent and the current representation vectors are analyzed using LSTM (Long Short-Term Memory) in order to encode them in one vector representing the page pair and to reduce the vector dimensionality. The choice of LSTM in this phase is to introduce the page sequence to the model. Using a dense layer, the pair representation vector is classified as a rupture or a continuity. The training of this model consists of two steps: Doc2Vec model training in order to learn the page representation vectors, and LSTM and Dense weights training to learn the stream segmentation task (see Figure 2).

3.2 GRU

GRU (Gated Recurrent Unit) is a recurrent neural network. Unlike the previous model, it learns words embeddings and the segmentation task simultaneously. We could use LSTM instead of GRU layers, but we got the same performance. So, we decided to keep GRU for its simplicity.

The GRU-based model proposed here learns words embeddings by analyzing the pair pages independently in two layers of GRUs. The GRUs analyze the current (GRU_CP) and previous page (GRU_PP) words embeddings. In order to obtain a page representation vector, we use an aggregation operation: summation or averaging. Since the GRU analyzes the text word by word, so, the output of the word$_i$ will depend on all the previous words in the sequence. So, the sequence level information is encoded in the output of the GRU. The idea behind the path separation is to learn different layer weights for the current and precedent page paths in order to treat their words differently. This separation will add information con-

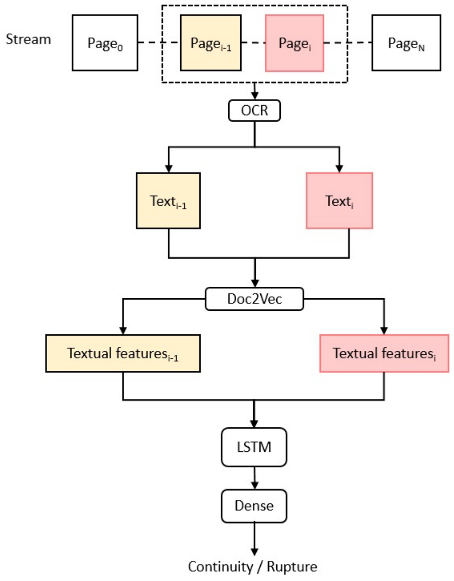


Figure 2: Doc2vec + LSTM Model architecture.

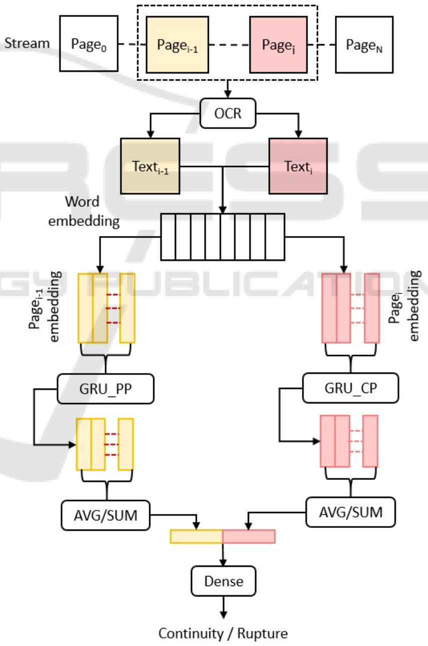


Figure 3: GRU Model architecture.

cerning the word source (i.e. current or precedent page). The classification is performed using the dense layer on the pair vector, obtained by concatenating the page pair vectors (see Figure 3).

3.3 AGRU

This model is an improvement of the previous one. Because the GRU model aggregates page word embeddings to obtain the page representation vector, all the page words are performed identically even those that mark rupture or continuity. To fix the GRU issue, we propose an attention mechanism that allows the model to detect words that indicate a relationship between the pages pair.

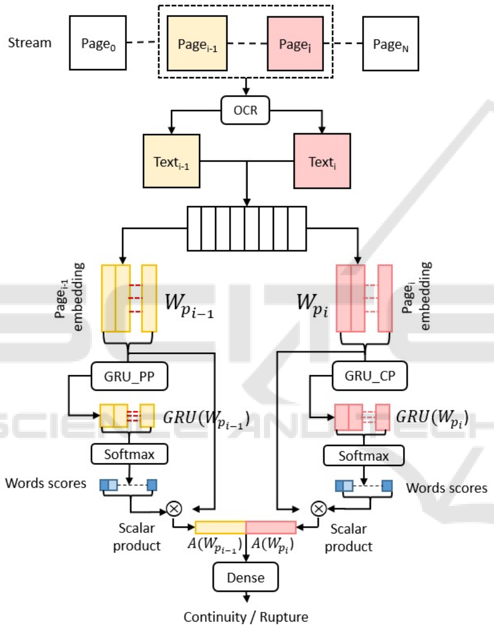


Figure 4: AGRU Model architecture.

The attention mechanism in this model is performed by applying a Softmax activation function on each output of the GRUs in order to obtain probabilities for each word. Once the scores are calculated, we calculate a page vector by performing a product between scores and their embedding. The concatenation of $page_{i-1}$ and $page_i$ vectors represent the input pages pair which will be classified by the dense layer (see Figure 4).

4 DATASETS DESCRIPTION

In this section we will describe the used datasets to train and evaluate the proposed models. In addition, we will explain how to prepare the page pairs for training. We have to precise at this point that the considered pages are scanned images ocrized using Tesseract open source Google OCR. We can notice that in this study, the quality of the OCR has not been taken into consideration. This means that we have processed the texts with their possible OCR errors.

4.1 READ-Corpus

This is a private collection of multi-page document images. It consists of 898 documents (3819 pages) of type invoice (INV), medical article (MA), official journal (OJ), conference article (CA) and training councils documents of the University of Lorraine (TC). The Table 1 shows the distribution by number of documents, number of pages, and average number of pages per document for all the classes.

Table 1: READ-Corpus Description.

Classe	MA	INV	TC	CA	OJ
Document count	68	216	140	181	293
Page count	437	216	893	1536	737
Average page number	6.4	1	6.3	8.4	2.5

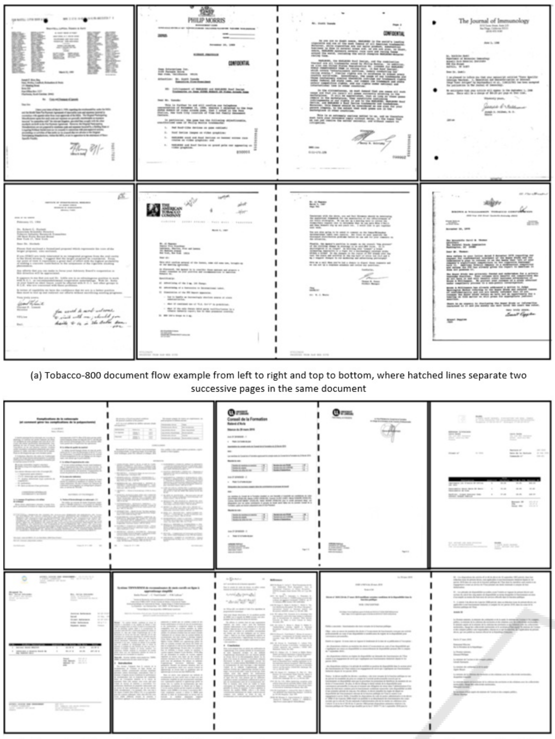
4.2 Tobacco-800

This public dataset (Zhu and Doermann, 2007) consists of 1290 scanned pages of realistic documents for research in image analysis of documents. These documents were collected and digitized by tobacco industry organization in the United States. This database contains 743 multi-page documents (1.7 pages per document on average) such as letters and invoices. Tobacoo is a complex base, it has been used in signatures and logos localization in (Lewis et al., 2006). The Figure 5 shows some documents issued from Tobacco-800 and from Read-Corpus.

4.3 Pairs Preparation

At first, we split the documents into three sets: 60% for training, 20% for validation, and 20% testing. For READ-Corpus, the splitting is done by balancing the number of documents per class in each set (see Table 2). Since the Tobacco-800 database is not labeled by document class, it has been split by document (see the Table 3).

As our segmentation models take the pages pairs as input, we have created pages pairs from the previously mentioned databases. On the one hand, pairs that represent continuity has been created from two consecutive pages of a document. On the other hand, ruptures are created from pages pairs where the first



(a) Tobacco-800 document flow example from left to right and top to bottom, where hatched lines separate two successive pages in the same document

(a) Example of document flow of READ-Corpus where the successive pages of the same document represent the first and the last page

Figure 5: Examples of some documents from Tobacco-800 (a) and READ-Corpus (b).

Table 2: READ-Corpus splitting result.

Set		MA	INV	TC	CA	OJ
Train	Docs count	40	192	84	108	175
	Pages count	257	129	540	902	438
	Page average	6.4	1	6.4	8.3	2.5
Validation	Docs count	13	43	28	36	58
	Pages count	79	43	177	312	145
	Page average	6	1	6.3	8.6	2.5
Test	Docs count	15	44	28	37	60
	Pages count	101	44	176	322	154
	Page average	6.7	1	6.2	8.7	2.5

Table 3: Tobacco-800 splitting result.

Set	Docs count	Pages Count	Pages average
Train	531	785	1.4
Validation	177	244	1.3
Test	178	261	1.4

page corresponds to the last page of a document, while the second page represents the first page of another document. The pairs are generated so that the number of continuities is equal to the number of ruptures in order to obtain balanced datasets (see the Table 4).

Let W_p be the page p words embeddings matrice. Let $h_t(W_{p_t})$ be the output of the GRU at the instance t using as input the Word embedding W_{p_t} of the word at the position t belonging to the page p. $GRU(W_p)$ is the result of passing through all the words of the page p by the GRU layer.

$$GRU(W_p) = [h_0(W_{p_0}), h_1(W_{p_1}), ..., h_0(W_{p_n})]$$

where n is the max page length in the dataset.

Let $Attention(W_p)$ be the result of applying the attention mechanism on the page p which could be the current or the precedent page.

$$Attention(W_p) = Softmax(GRU(W_p)) \times W_p^{\intercal}$$

Table 4: Page pairs description.

Datasets	Class	Training set	Validation set	Test set
READ-Corpus	Continuity	1730	578	613
	Rupture	1730	578	613
Tobacco-800	Continuity	254	67	83
	Rupture	241	79	93

5 RESULTS AND EXPERIMENTS

5.1 Models Parameters Tuning

The Doc2Vec language models are trained on each dataset, in order to learn representations for each document page. The training parameters are given in the Table 5.

Table 5: Language model parameters.

Model	Epochs	Vector size	Window size	Learning rate
Doc2Vec	50	100	5	0.1

The learning rate and the window size values are provided by the Gensim framwork (Rehurek and Sojka, 2010). With regard to the number of iterations and the size of the vectors, their values have been experimentally established. We found that increasing vector size does not improve the performance of the segmentation model. In this respect, we chose 100 as vector size for the Doc2Vec models in order to have simple and efficient model.

As for the architecture training previously proposed in the section 3, a script was created to vary the parameters as the activation functions, the unit number and the dropout rate, for each model layer. Regarding the number of iterations, we used the EarlyStoping mechanism of the Keras framework (Chollet, 2015) which consists in stopping the learning if there is no improvement in validation set loss after 20 iterations. The optimizer used during the learning is Adam with a learning rate equal to 0.001. The Tables 6 and 7 summarize the best model configuration.

Table 6: Models parameters for the READ-Corpus database.

Model	Layer	Unit Nb	Dropout	Recur. dropout	Activation
Doc2Vec +LSTM	LSTM	64	0.4	0.4	ReLu
	Dense	1	-	-	Sigmoid
GRU	GRU PP	8	0.2	0.6	ReLu
	GRU CP	8	0.2	0.6	ReLu
	Dense	1	0.4	-	Sigmoid
AGRU	GRU PP	1	0.0	0.2	ReLu
	GRU CP	1	0.0	0.2	ReLu
	Dense	1	0.4	1	Sigmoid

Table 7: Models parameters for the Tobacco-800 database.

Model	Layer	Unit Nb	Dropout	Recur. dropout	Activation
Doc2Vec + LSTM	LSTM	64	0.4	0.4	ReLu
	Dense	1	-	-	Sigmoid
GRU	GRU PP	8	0.0	0.8	ReLu
	GRU CP	8	0.0	0.8	ReLu
	Dense	1	0.4	-	Sigmoid
AGRU	GRU PP	1	0.4	0.6	ReLu
	GRU CP	1	0.4	0.2	ReLu
	Dense	1	0.4	1	Sigmoid

5.2 Results

Now, let's move to the evaluation of our models. We remind in the following the calculation formulas:

- TP (True Positive): sample number belonging to class A and classified in class A by the model.
- FP (False Positive): sample number belonging to class A and classified in class B by the model
- FN (False Negative): sample number belonging to class B and classified in class A by the model.
- TN (True Negative): sample number belonging to class B and classified in class B by the model

On the basis of these elements, we define the following quantities:

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}}$$

$$Recall = \frac{\text{TP}}{\text{TP+FN}}$$

$$Precision = \frac{\text{TP}}{\text{TP+FP}}$$

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision + Recall}}$$

The Table 8 summarizes the performances of our proposed models on the test set of READ-Corpus and Tobacco-800, by following the protocol of experimentation explained in the subsection 5.1.

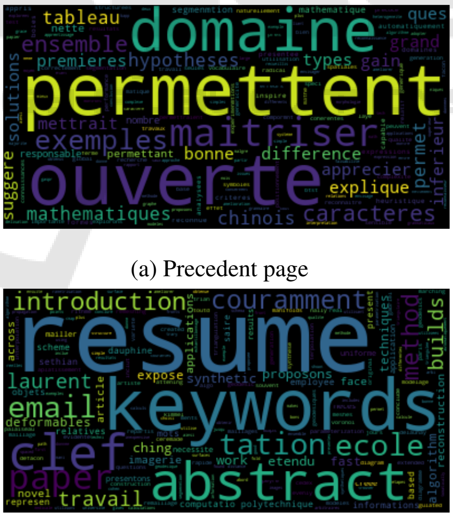
Table 8: Models performance.

Modele	READ-Corpus		Tobacco-800	
	Accuracy	F1	Accuracy	F1
Doc2vec + LSTM	0.94	0.94	0.83	0.83
GRU	0.93	0.93	0.88	0.88
AGRU	0.96	**0.96**	0.90	**0.90**

According to the Table 8 and the confusion matrices in Table 11 and Table 12, the AGRU outperforms the Doc2Vec + LSTM and GRU. The Table 9 reports the state of the art method performance among different datasets. According to this table, the AGRU model has given satisfying performances.

The Figures 6 and 7 represent the word clouds of the weights calculated by the attention mechanism from two pages pairs taken from READ-Corpus and Tobacco-800 respectively.

The word cloud illustrated in the Figure 6 concerns a READ-Corpus pair of whose previous page is of the TC class, whereas the current page belongs to the CA class. The attention mechanism has given significant weights to the words such as “resume”, “keywords”, “abstract” that indicate a rupture for the CA class (see Figure 6b).



(a) Precedent page

(b) Current page

Figure 6: A cloud indicating a rupture for the CA class in READ-Corpus.

The Figure 7 represents the weights calculated by the attention mechanism for another pair representing a rupture. In the case of this pair of pages, the AGRU model manages to detect a rupture. Since the current page is part of the class letter, the attention mechanism has given significant weights to the words that

Table 9: Comparative table of segmentation models.

Author	Model	Input	Dataset	Accuracy	F1
Daher & Belaïd (Daher and Belaid, 2014)	SVM	Text	Private	-	80%
	MLP	Text	Private	-	83%
Gallo & Noce (Gallo et al., 2016)	CNN + MLP	Image	Private	97%	-
Hamdi & Coustaty (Hamdi et al., 2018)	Doc2-Vec	Text	Private	84%	-
Wiedemann & Heyer (Wiedemann and Heyer, 2017)	CNN Text + CNN Image	Text + Image	Archive22k (private)	93%	-
			Tobacco-800	91%	-
Our model	AGRU	Text	READ-Corpus	96%	96%
			Tobacco-800	90%	90%

Table 10: GRU.

		Predicted	
		C	R
Actual	C	597	16
	R	56	557

(a) Doc2Vec + LSTM.

Table 11: Models confusion matrices on READ-Corpus database where C: Continuity and R: Rupture.

		Predicted	
		C	R
Actual	C	592	21
	R	68	545

		Predicted	
		C	R
Actual	C	608	5
	R	43	570

(a) AGRU

Table 12: Models confusion matrices on Tobacco-800 database where C: Continuity and R: Rupture.

		Predicted	
		C	R
Actual	C	71	12
	R	17	76

(a) Doc2Vec + LSTM.

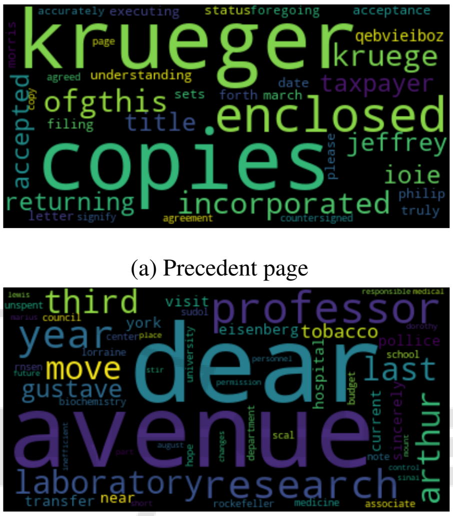
		Predicted	
		C	R
Actual	C	67	16
	R	5	88

(b) GRU.

		Predicted	
		C	R
Actual	C	74	9
	R	8	85

(c) AGRU

represents the beginning of the page of this class of document, such as "dear", "avenue" (see Figure 7b).



(a) Precedent page

(b) Current page

Figure 7: A cloud indicating a rupture in Tobacco-800.

6 CONCLUSION AND PERSPECTIVES

In this paper, we described segmentation models based on deep learning. At first, we proposed Doc2Vec + LSTM based on language models Word2Vec. The training of this model was carried out in two stages: the training of the language model; the training of the different layers for the global model. In general, Doc2vec is used for text classification and sentiment analysis tasks. The training of these language models is not oriented segmentation task because it is an unsupervised training. This led us to propose the models GRU and AGRU which learns the segmentation task and the word embedding simultaneously.

The evaluation results show that the AGRU is the best of the models using textual information. This is due to the attention mechanism that reinforces the segmentation task by giving high scores to important words that represent the first page of a document.

Given the lack of administrative databases at public reach, we will propose to develop a synthetic administrative document generator in order to push the search into the document flow segmentation task. Since our AGRU model deals with lexical entities only, we will propose to combine the AGRU model with models capable of identifying named entities, dates, identifiers, physical and logical characteristics, and factual information.

REFERENCES

Chollet, F. (2015). Keras. Technical report, https ://keras.io.

Daher, H. and Belaid, A. (2014). Multipage administrative document stream segmentation. In *International Conference on Pattern Recognition (ICPR)*, pages 966–971.

Furcy, D. and Koenig, S. (2005). Limited discrepancy beam search. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 125–131.

Gallo, I., Noce, L., Zamberletti, A., and Calefati, A. (2016). Deep neural networks for page stream segmentation and classification. In *International Conference on Digital Image Computing : Techniques and Applications (DICTA)*, pages 1–7.

Gordo, A., Rusinol, M., Karatzas, D., and Bagdanov, A. (2013). Document classification and page stream segmentation for digital mailroom applications. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 621–625.

Hamdi, A., Coustaty, M., Joseph, A., D'Andecy, V. P., Doucet, A., and Ogier, J. (2018). Feature selection for document flow segmentation. In *13th IAPR Workshop on Document Analysis Systems (DAS)*, pages 245–250.

Hamdi, A., Voerman, J., Coustaty, M., Joseph, A., d'Andecy, V. P., and Ogier, J. (2017). Machine learning vs deterministic rule-based system for document stream segmentation. In *IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 5, pages 77–82.

Harley, A., Ufkes, A., and Derpanis, K. (2015). Evaluation of deep convolutional nets for document image classification and retrieval. In *13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995.

Karpinski, R. and Belaid, A. (2016). Combination of structural and factual descriptors for document stream segmentation. In *12th IAPR Workshop on Document Analysis Systems (DAS)*, pages 221–226.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., and Heard, J. (2006). Building a test collection for complex document information processing. In *Proc. ACM SIGIR Conf.*, pages 665–666.

Meilender, T. and Belaid, A. (2009). Segmentation of continuous document flow by a modified backward-forward algorithm. In *Document Recognition and Retrieval*, pages 1–10.

Noce, L., Gallo, I., Zamberletti, A., and Calefati, A. (2016). Embedded textual content for document image classification with convolutional neural networks. In *Proceedings of the 2016 ACM Symposium on Document Engineering (DocEng)*, pages 165–173.

Rehurek, R. and Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50.

Schmidtler, M. and Amtrup, J. (2017). Automatic document separation: A combination of probabilistic classification and finite-state sequence modeling. In *Natural Language Processing and Text Mining*, pages 123–144.

Wiedemann, G. and Heyer, G. (2017). Page stream segmentation with convolutional neural nets combining textual and visual features. In *CoRR, abs/1710.03006,2017*.

Wiki, A. (2017). A.i. wiki, a beginner's guide to important topics in ai, machine learning, and deep learning: A beginner's guide to word2vec and neural word embeddings. In *https://skymind.ai/wiki/word2vec*.

Zhu, G. and Doermann, D. (2007). Automatic document logo detection. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 864–868.