

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

MASTER PROJECT

Historical newspaper semantic segmentation using visual and textual features

Author:
Raphaël BARMAN

Supervisors:
Dr. Maud EHRMANN
Dr. Simon CLEMATIDE
Sofia ARES OLIVEIRA

Professor:
Prof. Frédéric KAPLAN

Digital Humanities Laboratory
EPFL

June 21, 2019

EPFL

Abstract

Mass digitization and the opening of digital libraries gave access to a huge amount of historical newspapers. In order to bring structure into these documents, current techniques generally proceed in two distinct steps. First, they segment the digitized images into generic articles and then classify the text of the articles into finer-grained categories. Unfortunately, by losing the link between layout and text, these two steps are not able to account for the fact that newspaper content items have distinctive visual features. This project proposes two main novelties. Firstly, it introduces the idea of merging the segmentation and classification steps, resulting in a fine-grained semantic segmentation of newspapers images. Secondly, it proposes to use textual features under the form of embeddings maps at segmentation step. The semantic segmentation with four categories (feuilleton, weather forecast, obituary, and stock exchange table) is done using a fully convolutional neural network and reaches a mIoU of 79.3%. The introduction of embeddings maps improves the overall performances by 3% and the generalization across time and newspapers by 8% and 12%, respectively. This shows a strong potential to consider the semantic aspect in the segmentation of newspapers and to use textual features to improve generalization.

Acknowledgements

I would like to thank my supervisors, Maud Ehrmann, Sofia Ares Oliveira and Simon Clematide for their continuous guidance, advice and proofreading.

I am also particularly grateful to Julien Nguyen Dang, Maud Ehrmann, Simon Clematide and Tobias von Waldkirch who annotated part of the data.

Finally, I am deeply thankful to my girlfriend for her support and her precious proofreading who spared the readers many convoluted sentences.

Contents

Abstract	iii
Acknowledgements	v
1 Introduction	1
2 Background information and theory	3
2.1 Semantic segmentation	3
2.1.1 Natural image semantic segmentation	3
2.1.2 Document image semantic segmentation	4
2.2 Deep learning for Image processing	5
2.2.1 Neural networks	5
2.2.2 Convolutional Neural Networks	6
2.2.3 Transfer learning	7
2.2.4 Residual networks	7
2.2.5 Segmentation architecture	8
2.3 Text embeddings	9
2.3.1 Discrete embeddings	10
2.3.2 Continuous embeddings	10
2.4 End-to-end learning	11
2.5 Multimodal learning	12
2.6 Text embeddings map	12
3 Method	15
3.1 Hypotheses	15
3.2 Dataset	16
3.2.1 Newspapers	16
3.2.2 Classes	17
3.2.3 Statistics	19
3.3 Text embeddings map implementation	20
3.4 Model architecture	21
3.4.1 Concatenation of the embeddings map	21
3.4.2 Size and nature of the embeddings	21
3.5 Evaluation methodology	24
3.5.1 Mean Intersection over Union	24
3.5.2 Precision and Recall	25
3.6 Model variance	26
4 Experiments	29
4.1 Experimental setup	29
4.2 Baseline model	31
4.3 Embeddings experiments	31
4.3.1 Embeddings type and input level	32

4.3.2	Blank embeddings	36
4.3.3	Embeddings transformation	40
4.3.4	Discussion	43
4.4	Generalization experiments	43
4.4.1	Amount of training data	44
4.4.2	Generalization across time	47
4.4.3	Generalization across newspaper	50
4.4.4	Discussion	52
5	Discussion and conclusion	55
5.1	Discussion	55
5.2	Contributions	56
5.3	Limits	56
5.4	Improvement and future work	56
A	Evaluation interface	59
	Bibliography	63

1 Introduction

Historical newspapers are an important primary source for historians. Indeed, they contain information on local and minor events, but also on international and major ones. Newspapers also reflect the way these events were perceived and what were the main social concerns at the time of publication. Newspapers are often published on a daily or weekly basis and are most of the time regional. Hence there is a huge amount of them and they have usually been archived.

These past few years, important efforts to bring these newspapers archives into a digital form were undertaken through mass digitization. Moreover, progresses in optical character recognition (OCR) and optical layout recognition (OLR) have made these archives more accessible and searchable. Since the only entry point to this mass of information is keyword search and since metadata available is scarce, it is difficult to find the right information. It is even more difficult to make a quantitative analysis of newspapers.

The current OLR systems only focus on very basic categories. They can detect articles, advertisements, and images. However, newspapers have richer content than these basic separations and there exist more specific content items that are common across different newspapers and through time. For example, nearly all newspapers have a weather forecast and an obituaries section. Finding and extracting these elements would be interesting from different perspectives. For historians, it would allow for both finer grained searches and quantitative analysis. For example, when considering a particular content item, it would be possible to find out when it appears or disappears and to analyze the changes of language it undergoes. For the natural language processing researchers, having this information would also be of interest for improving further processing of the newspapers' text. For example, it could allow to filter out sections of the newspapers that are known to have especially noisy OCR, such as stock exchange tables.

The task of article segmentation is difficult, particularly because content items change in term of layout and language between newspapers and across time. The approaches that are currently trying to tackle these difficulties are part of two distinct research fields. The first one, document image segmentation, comes from the domain of image processing. It segments newspapers images in generic categories like articles, in order for them to be further processed. The second approach is text classification and comes from the domain of natural language processing. Once the text is extracted from the segmented image by an OCR process, it is classified into different categories. One flaw of these approaches is that the link between the layout and the text is lost. Indeed, one approach focuses only on the visual aspect and the other one only on the textual features. This is quite problematic in the case of newspapers since both layout and text can be informative of the type of content item.

This project is a first step to reconcile the two research fields. It tries to segment the images of the newspapers and to semantically qualify the detected segments. This means that, more than detecting an article, it will be able to directly detect its precise category (e.g. "Weather forecast"). Moreover, since the text is a central element of newspapers, it will be used jointly with the image. Hence, the goal of this project is to evaluate how the introduction of textual features influences the results of semantic segmentation. Indeed, by using both modalities at the same time, this project tries to counterbalance the ambiguities of the visual with the

textual and vice-versa. The visual information should help recognize the layout of a particular category of item, while the textual information should help to make the system more robust and allow for a generalization of the results. Indeed, it will have to be taken into account that some content items in newspapers can be confused and that the layout evolves through time and across newspapers.

This report is outlined as follows. First, Section 2 gives information on the main theoretical foundations this project is based on. Then Section 3 states hypotheses and explain the method used to answer them. Section 4 introduces and discusses the different experiments used to explore the goal of this project, first by exploring different configurations of mixes of visual and textual features, then by finding how they are able to generalize across time and newspapers. Finally, Section 5 discusses the findings and contributions, analyze the limits of the project and proposes improvement and potential future work.

2 Background information and theory

This chapter gives a general background on the methods used in this project. Section 2.1 introduces the notion of semantic segmentation and how it can be applied to natural and document images. Section 2.2 gives a general overview of how deep learning is used for image processing and introduces architectures for semantic segmentation. Section 2.3 presents the notion of text embeddings and different implementations. Section 2.4 explains the notation of end-to-end learning and Section 2.5 introduces the concept of using several modalities in a model. Finally, Section 2.6 presents the notion of text embeddings maps that allow to use both textual and visual features in the same model.

2.1 Semantic segmentation

Semantic segmentation is an essential step towards understanding images. Indeed, having information about the what and the where of the content of an image is crucial to many tasks, such as medical imagery [1], document layout recognition [2] [3] and autonomous driving [4].

Image segmentation is the task of finding regions of interest and detecting structure. Semantic segmentation adds the idea that each segment is semantically qualified. For example, the segment is qualified with a particular brand of car in a natural image of a street or a particular type of ornament in a document image of an ancient manuscript. Having this granularity of information is very helpful for downstream tasks. For example, it becomes possible to apply natural language processing on one particular typology of newspapers article.

Semantic segmentation is a more complex task than image classification, because rather than treating the entire image, it is treated either by general regions (bounding boxes for examples) or at the pixel level.

2.1.1 Natural image semantic segmentation

Semantic segmentation research mainly focuses on natural images. These are here understood as photography of the real world, this is in opposition to artificial images which are either computer made or photography of man-made document. Various improvements in this domain were made in the past few years, in particular with the introduction of the fully convolutional network [5], the pyramidal pooling module [6] and the atrous convolution block [7]. The most famous datasets include the PASCAL Visual Object Classes (VOC) [8] and Microsoft Common Objects in Context (COCO) [9] which have a variety of classes ranging from vehicles to animals and to everyday objects. There is a large diversity of other datasets and applications ranging from biomedical imagery, to urban street scene analysis for autonomous driving [4].

2.1.2 Document image semantic segmentation

Document image segmentation is arguably easier than natural images segmentation. Indeed, apart from the occasional warp or noise on the images, documents are much more regular. Indeed, the number of variations that they have is more limited than a natural scene, mainly because of the physical constraint of the page. This number of variations is however still important because of the variety of documents and the different layouts they have. It is thus often difficult to generalize without losing accuracy. This is true with newspapers where the same semantic item can be presented very differently depending on the period or the newspaper.

Segmentation is most of the time only one part of a document content extraction pipeline. It usually occurs after basic pre-processing and before classification and other processings such as optical character recognition. Hence, it mainly focuses on segmenting “low-level” features, either part of the physical layout such as text, graphics, separators or part of the logical layout such as headers, footnotes and main body.

There are several approaches to document image segmentation. The survey in [10] gives a good overview of the different techniques. They can mainly be separated in two categories: top-down and bottom-up.

Top down approaches start from the whole page and try to partition it in smaller segments recursively. The most well known techniques are the Run Length Smoothing Algorithm (RLSA) [3] and the X-Y cut algorithm [11]. They both rely on finding separation between segments using the amount of printed pixel horizontally and vertically. These methods were the first to be developed and mainly aim at segmenting a specific layout, thus being usually used without training. Moreover, most of the approaches rely on the fact that the documents have a “Manhattan” layout, i.e. that every segment can be separated using only vertical and horizontal lines.

Bottom-up approaches start from small components such as pixels, connected components or “patches” and try to aggregate them into bigger components also in a recursive manner. They can thus better adapt to local variations in the document and segment a larger variety of layouts, particularly ones that are not “Manhattan”. They however have a higher number of parameters and hand-crafted features that are quite difficult to tune and may need to be trained on a large amount of data.

These last few years, new approaches that are neither top-down nor bottom-up have appeared. They are based on deep neural networks and thus do not rely on the previously hand crafted features and expert knowledge, but are able to learn a representation suited for the task by themselves.

A recent analysis of the use of deep learning in document analysis [12] showed an increase in term of scientific papers whose subject were neural network in international document analysis conferences, such as the 2016 International Conference on Pattern Recognition (ICPR2016) and the 2017 International Conference on Document Analysis and Recognition (ICDAR2017). However their usage is not completely exclusive and other methods still continue to be competitive on challenges such as the ICDAR2017 Competition on Recognition of Documents with Complex Layouts [13] where no participants used neural networks.

A general trend in document segmentation is that all the elements segmented are very low level, such as character or lines. Even when treating higher level elements, such as in the ICDAR2017 Competition on Page Object Detection [14] where formulae, tables and figures are segmented, these elements are still very generic, and do not really take into account the semantic dimension of the segments. This is somewhat problematic as outlined in [12] since

documents are produced and used by human, they will have some visual rules that are the results of long evolution. There is thus a strong potential to segment higher level elements, since these rules are lost when segmenting low level elements, since the layout is flattened when going into the one-dimensional level of text.

The current use case of those higher level semantic features in an end-to-end fashion is in segmenting documents of administrative purposes, such as invoices, where the items are of a semantic level (order id, level, date, etc.), c.f. for example [15] and [16].

The segmentation of newspaper is no exception and the highest level elements that are segmented are articles, titles and sometimes advertisements and images. Early works all used hybrid approaches, either favoring the RLSA algorithm as in [17] and [18] or the X-Y cut algorithm as in [19] in order to find lines of texts and finally cluster them into articles. One of the most recently published work by Meier et al. [20] made use of a neural network for the segmentation, but only treated the binary article or not-article case. There is thus no previous research that tried to segment more semantically rich content.

2.2 Deep learning for Image processing

2.2.1 Neural networks

Since their beginning in the 40s and 50s [21] [22], neural networks have come a long way. They are inspired by how the biological neural network works, in the sense that they are a succession of connected artificial neurons that can send signal to each other such as synapses do in the brain.

The most basic mathematical formulation is the one of a linear layer followed by a non linear activation, this will be referred to as a node: $y = \phi(\mathbf{A} \cdot \mathbf{x})$.

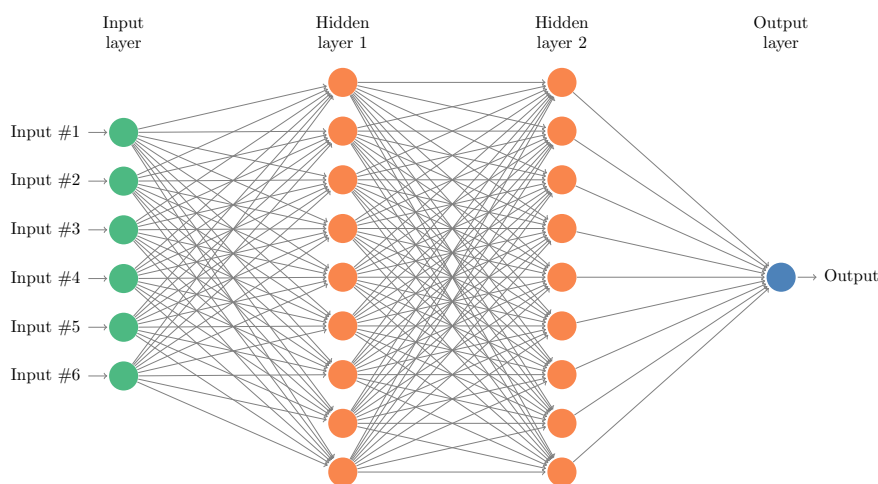


FIGURE 2.1: A fully connected feed-forward neural network. It has two hidden layers of each 9 nodes. It can be seen that the number of connections is quadratic with the number of nodes and hidden layers.

Popular non linearities are for example the sigmoid function $\phi(x) = \frac{1}{1+e^{-x}}$ or the rectified linear function (ReLU) $\phi(x) = \max(0, x)$. By combining these nodes in layers in a acyclic fashion, meaning that each output of a layer is fed as the input of the next layer, we obtain a feed-forward network. When each node of the layer is connected to every node of the next

layer, this is called a fully-connected layer which consist of one of the most basic block of neural networks. An example is shown on figure 2.1.

The combination of these simple building blocks have interesting mathematical properties that allow them to approximate any continuous function of \mathbb{R}^N on a bounded domain, given enough of them and the right weights (the matrix A in the linear part of the previous equation) [23]. In practice, the main difficulty lies in training a model to have the best possible approximation of the weights. This optimization is done through the backpropagation algorithm that uses gradient descent and the chain rule to compute updates for every weights of the network in order to make it better approximate the target function.

2.2.2 Convolutional Neural Networks

A drawback of a fully connected network is that it has lots of weights that need to be updated. Particularly when dealing with images the number of parameters quickly explodes. For example with an image of 300 pixels in each dimension and 3 color channels, connected to a hidden layer of 128 nodes would result in $300 \cdot 300 \cdot 3 \cdot 128 \simeq 34.6 \cdot 10^6$ connections.

A solution to this problem is to use the fact that images exhibit most of the time a lot of local spatial relationships. Indeed, neighboring pixels should not be affected by their position inside the image, but only by their relationship to one another. This is something that has been long used in traditional image processing with the usage of convolution filters. These filters can outline various aspect of the underlying signal, for example they can smooth features by averaging or make differences more salient between neighboring elements by taking a “high-pass” filter.

In a convolutional layer, the idea is to learn such filters through gradient descent. It is usual to stack several filters inside one convolutional layer, each filter giving a new different feature map. An example architecture can be seen in Figure 2.2.

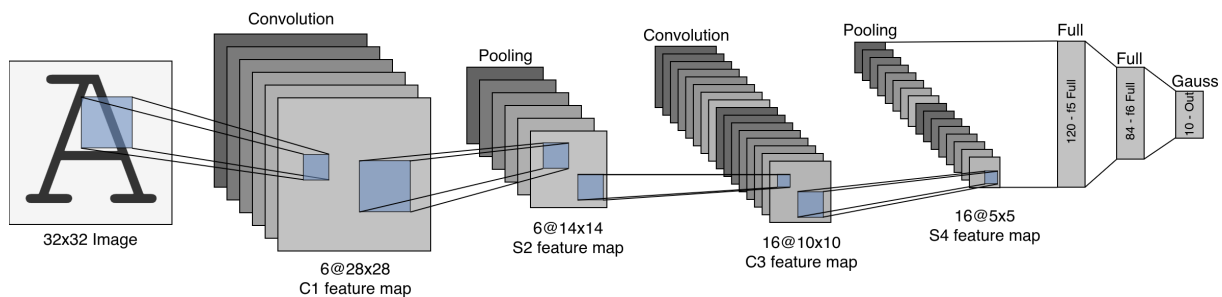


FIGURE 2.2: The “lenet-5” network [24], the architecture that popularized the usage of convolutional layers. The input image has only one feature map, the gray channel, then going through different convolutional layers, the number of feature maps increases while their sizes are reduced. Finally the resulting feature maps are flattened and go through fully connected layers for a final prediction. This is a very common pattern in convolutional neural networks.

The usage of convolutional neural network architecture was again made popular in 2012 by its high performance on a very complex problem. The ImageNet database [25] is a set of 15 millions high resolution images labeled according to 22’000 classes. The ImageNet Large-Scale Visual Recognition Challenge proposes a subset of 1000 classes with around 1000 images per class. Before 2012, the progress on this task was made using “traditional” image processing methods and the scores increased slowly and steadily. In 2012, the AlexNet architecture [26] was proposed. It made usage of graphical processing unit (GPU) in order to be able to train

what is called a *deep neural network* comprising 650'000 neurons and more than 60 million parameters, improving the current state of the art for ImageNet by more than 10.8 points.

2.2.3 Transfer learning

The combination of the three advances previously mentioned, namely the access to large databases of annotated images, the usage of *deep neural network architectures* and the possibility to train them using GPUs, made neural network popular again, particularly in the domain of image processing. Moreover, models that were trained on ImageNet were found to be able to “transfer” very well to other image processing tasks that had less training data.

The idea is to use a *pre-trained* network on the ImageNet task, i.e. to start from the weights of the supervised pre-training rather than random weights as it is usually done. Then the network is “fine-tuned”, either by using a smaller learning step or by “freezing” some network layers so that the weights do not change. This approach has shown to work on various tasks where the amount of training data is scarce. For example in medical image classification [27], semantic segmentation [5], human pose estimation [28]. This pre-training has also shown that the same pre-trained architecture could be used for a variety of documents and tasks [29], achieving very good results for a simple segmentation task with only 100 training examples.

This project has a relatively small amount of annotated data, particularly for some layouts that are only present a few times. Thus every network used will make use of pre-trained weights on Imagenet.

2.2.4 Residual networks

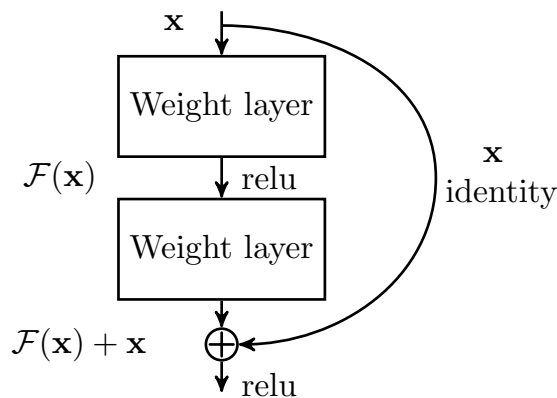


FIGURE 2.3: A residual block as presented in [30]. The weight layers and ReLU blocks are what is usually present in a neural network, the novelty is the addition of the skip connection, that adds to the output of the weight layer what was at the input of the block.

Another important network architecture that needs to be presented is the residual network [30]. The network makes usage of “shortcut connections” shown in Figure 2.3. More formally, rather than fitting the desired mapping $\mathcal{H}(x)$, the residual mapping $\mathcal{F}(x) = \mathcal{H}(x) - x$ is instead fitted, thus the mapping $\mathcal{H}(x)$ becomes $\mathcal{F}(x) + x$. The motivation behind this kind of block is the degradation problem, i.e. that usually the training error increases as the depth increase. However, if the added layers could be identity layers, then the error of the deeper model should

always be less or equal than its shallower counterpart. The residual block makes it easier to make this identity mapping. Indeed, it is enough to set the weights to zero, whereas before the mapping $\mathcal{H}(x) = x$ was not necessarily trivial to obtain. The authors, using these new blocks, managed to train networks deep between 18 and 1000 layers, with very good results at lower depth, but without significant improvement at very high depth. The best trade-off between depth and performance for ImageNet is at 50 layers, this model is often referred as Resnet-50 and will be used in this project.

2.2.5 Segmentation architecture

Several architectures have been proposed through the years for semantic segmentation of images. The focus here is only on architecture that perform pixel perfect segmentation, meaning that region proposal networks (RPN) are not considered. Mainly because the architecture segmenting at the pixel level are the most used in document image processing, whereas the RPN are mainly used on natural images.

Fully Convolutional Network

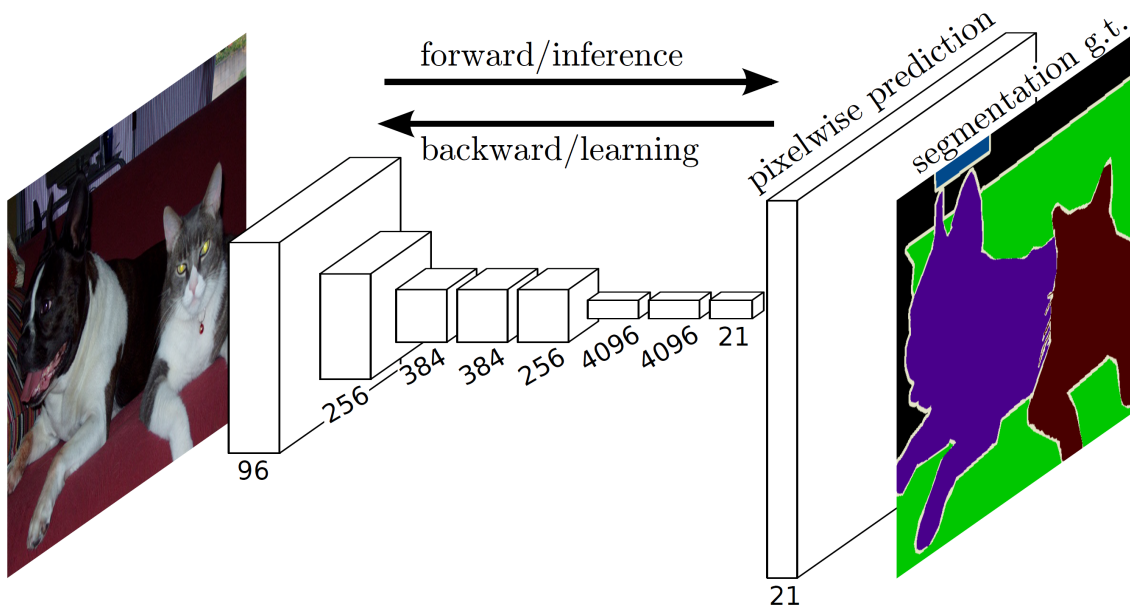


FIGURE 2.4: The fully connected network architecture [5]. The first part of the network is a VGG16 [31] that was pre-trained on imagenet and fine-tuned for semantic segmentation of natural images. Image credits: Long et Al.

The idea behind the fully convolutional network [5] is to take a network pre-trained on ImageNet and to replace the last fully connected layers by 1×1 convolutional filters. This makes the network fully convolutional as the final layers are not flattened and are still three-dimensional feature maps. Because the network reduces the image by max-pooling, the final result is usually smaller than the original image, but by upsampling it, it is possible to recover a pixelwise prediction of the same size as the input. This architecture is illustrated in figure 2.4. The main advantage of this architecture is that it is able to leverage the power of pre-training, since few changes need to be made to go from a classifying network to a segmenting one. Moreover, it works on images of arbitrary sizes, since the network is entirely convolutional.

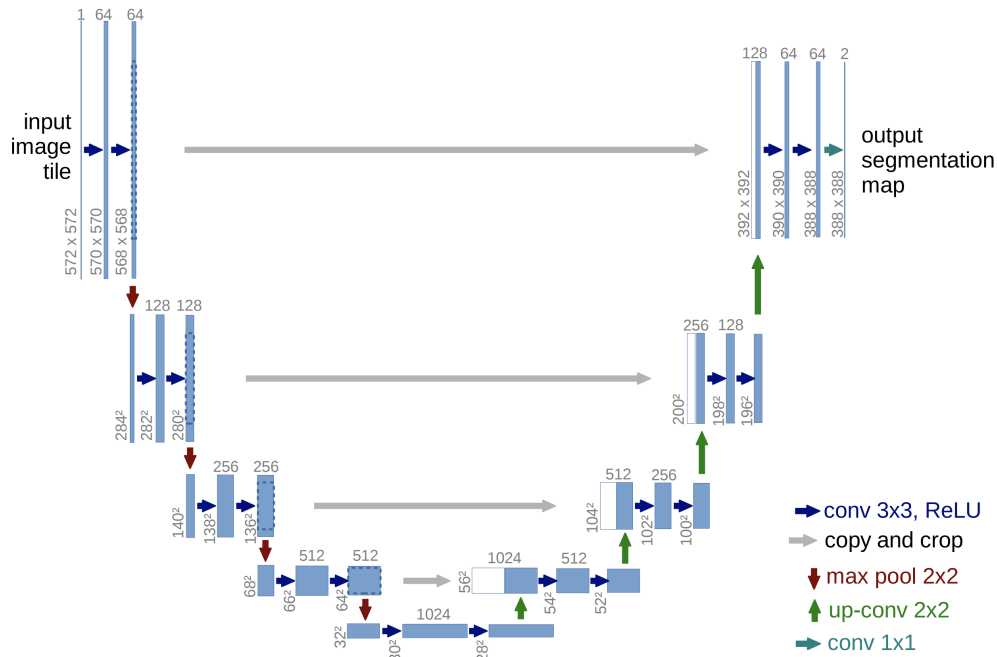


FIGURE 2.5: The U-net architecture [1]. It introduces copy connection that helps keeping high resolution features. Image credits: Ronneberger et Al.

U-net

The second network presented here is an evolution of the fully convolutional network. The “U-net” [1] is an architecture created for the segmentation of biomedical images, it is presented in Figure 2.5. This architecture distinguishes itself from the fully convolutional network in three points. First of all, its encoding part (left side of the network) is much less deep and is not pre-trained. Secondly, high resolution feature maps from the encoding part are concatenated to the decoding part (right side of the network) to help with the localization. Finally, the decoding part uses more convolutions in order to learn more precise features from the concatenation.

dhSegment

dhSegment [29] is an architecture based on U-net that is mainly used to segment document images. This network uses a pre-trained Resnet-50 in the same way the fully connected network used a VGG16 and also has supplementary convolutional layers in the decoding part of the network as introduced in the U-net. The architecture is presented in more details in section 3.4. It was developed with the purpose of being generic enough to be able to work on a variety of document image segmentation tasks, ranging from page and lines detection to the detection of ornaments. It showed to be very competitive on a variety of tasks, using the same architecture.

2.3 Text embeddings

Text usually cannot be used directly in neural networks or in other machine learning models, it first needs to be mapped to a numerical representation.

This means mapping each unit of a text, that can be considered at different levels, for example at the character or word level, to a numerical vector. Then these vectors can directly be used as a numerical representation of the considered unit by a model.

2.3.1 Discrete embeddings

The most basic numerical representation is “one-hot encoding”, which makes use of a vocabulary of tokens and creates a vector of the length of the vocabulary with a value of 1 if the token is in that position and 0 otherwise.

The problem with this approach is that the vector size grows linearly with the vocabulary size, which makes it quickly unmanageable when using large corpora. Moreover this approach does not give any information about the similarity of two tokens, since their vectors cosine similarity will always be zero.

2.3.2 Continuous embeddings

Another approach is to have a fixed sized vector that contains continuous values instead of discrete ones. An embedding is then characterized as good if two words that are semantically close are also close in the vector space. For example, the cosine similarity of “rain” and “drizzle” should be high, whereas the cosine similarity of “rain” and “rate” should be small.

This approach has the advantage of having fixed size vectors that can capture similarity between tokens. However, these vectors need to be learned which is not always straightforward and requires large amount of data.

Word2Vec

Word2Vec presented in [32] uses a neural network to learn embeddings at the word level. This approach is based on the idea that words that occur in the same context have the same meaning [33].

This model comes in two flavors. The first one tries to predict a word given its surrounding, this approach is called continuous bag of words (CBOW). The second one tries to predict the surrounding given a word, this approach is called Skip-gram. The prediction is done on the basis of a shared embeddings representation for each word which is adjusted during the training of the model.

The embeddings can be learned in a semi-supervised manner, using data that is not annotated, but simply using text. This training is done on large corpora such as Wikipedia in order to have good quality embeddings. These embeddings are then usually used as input to other models to realize a variety of tasks, such as text classification or named entity recognition.

It is important to note that this model learns a general representation for each word, regardless of its context and meaning. So words that are homonymous such as “bank” as in the sides of a river or the organization that handles money or that are polysemous such as “newspaper” as in the periodically printed document or the organization that publishes them, have the same embedding.

Fasttext

Fasttext¹ is a library developed by Facebook AI Research. It is an implementation of subword information skip grams [34] that uses a similar idea to Word2Vec, but that works on n -grams of characters inside a word (a subword) and represents words as the sum of its subwords. This has shown to perform better than Word2Vec on a variety of tasks and particularly when dealing with other languages. An advantage of this method is that it can deal with words that are not directly inside the base vocabulary if they have matching subwords. This can be especially important when dealing with noisy text coming from OCR as with newspapers.

ELMo

ELMo presented in [35] computes context-dependent embeddings of words, meaning that two same words can have different embeddings depending on their surrounding words. This model uses recurrent neural network to be able to keep track of the context and also tries to predict the next word for a particular sequence. It computes it considering the context both in a forward and a backward pass and uses a concatenation and a weighted sum of hidden states of the recurrent neural network to compute the embeddings.

As previously mentioned, its main advantage is to take into account the context of a word when computing its embedding and thus be able to account for the homonyms and polysemies.

Flair

Flair² is a library developed by Zalando Research. It is an implementation of contextual string embeddings for sequence labeling [36]. It uses the same idea as ELMo, but at the character level. It has shown to perform very well on a variety of tasks and on different languages. It has the big advantage of not depending on a vocabulary of words, but on characters, making it more robust to out of vocabulary words and OCR mistakes.

2.4 End-to-end learning

End-to-end learning refers to training models that are entirely differentiable and are self-sufficient to perform a particular task. It means that the same architecture takes an input and processes it into the desired output using only differentiable modules, i.e. modules that will learn how to process the input. For example, a system that does scene text recognition, i.e. that recognize text in natural images, for example street signs, can be trained in an end-to-end manner. It will take as input the image, will segment it in order to extract the text area and will pass it through a character recognizer. If these two modules are differentiable, then the system can be trained in an end-to-end manner as in [37].

End-to-end learning is quite beautiful conceptually, since the model is trained as whole and thus every parameters needed to treat the task can learned using data. Moreover, it has shown to work in a variety of domain, such as image recognition [26], scene text recognition, autonomous navigation [38], playing video games [39] and image segmentation [5].

¹<https://research.fb.com/fasttext/>

²<https://github.com/zalandoresearch/flair>

2.5 Multimodal learning

A modality is the way in which something exists or is expressed. For example images and text are two different modalities. Multimodal learning is when the learning is done on more than one modality, which is usually how humans experience the world. Multimodal learning has been mainly used in the fields of audio-visual speech recognition [40], multimedia content indexing [41] and more recently media transcription [42]. The survey of Baltrušaitis et al. [43] gives a good overview on the topic. In particular, two advantages are put forward. Firstly, having multiple modalities observing the same phenomenon increases the robustness of the prediction. Secondly, multiple modalities can complete each other, for example when something is visible in one of the modalities and not in the other.

One way of creating multimodal models is to use joint representations, which can for example be simply the concatenation of two unimodal representations. In the case of a neural network, this representation can be further processed in order to better adapt to the task. For example, Frome et al. [44], used pre-trained representation of both images and text and merged them to learn in an end-to-end manner better visual representation using the textual ones.

2.6 Text embeddings map

One of the research question of this project is too see if a multimodal approach to document image segmentation has advantages compared to an unimodal one as explained in Section 2.5. However, in order to use visual and textual features at the same time, a mapping from the usual embeddings form of the text to a form compatible with images is necessary. Previous works have made a mapping from the textual information to a 2D representation of it. The idea is that even though text is often considered as uni-dimensional, its physical nature is very much bi-dimensional. Indeed, the layout in which the text is disposed is also informative. Being composed of bi-dimensional elements such as lines, paragraphs and tables. These element are often bringing an additional layer of information to the reader. For example, titles are written with large font and describe the text physically under them.

The idea of mapping textual information to a 2D representation has been first proposed in [45] and uses the same technique this work will use. The approach consists of using text embeddings as described previously and to map them with their original position on the image. The authors focus on what they call document semantic structure extraction. They try to segment high level classes, such as figures, tables, headings, captions, lists and paragraphs. The goal of using the textual features is to reduce some false positives, since visual features are not always sufficient to differentiate. For example a big font may not always be a heading, or dashes inside a text might not always indicate a list. The results show a slight improvement compared to an approach that does not use textual features, particularly if the OCR provided is of good quality. It is the only research that uses text embeddings directly as a feature map.

Chargrid introduced in [16] uses a map constructed in a similar fashion as in the previously mentioned approach, but at the character level and using a one-hot encoding. However, the authors specify that one could use an approach at the word level using embeddings, but that this was not adapted to their particular use case, extracting invoice information. The approach shows good improvement using only the textual features encoded in a 2D fashion and even better improvement using both visual and textual features.

Finally, the approach presented in [15] uses one hot encoded n-grams of characters also mapped in a 2D space in as similar manner. This work, which also tries to extract information of invoices, shows that having textual features in a 2D space improves some results over an approach that only operates on textual features in a 1D space.

3 Method

This section presents the methodology used in order to reach the goal of this project. Section 3.1 makes two hypotheses deduced from the state of the art. Section 3.2 presents the data used in this project. Section 3.3 explains how the textual and visual features can be mixed. Section 3.4 introduces the model architecture. Section 3.5 explains the different metrics used for evaluating the different experiments. Finally, Section 3.6 gives a general framework to deal with the high variance of neural networks.

3.1 Hypotheses

The goal of this project is to assess how textual features influence the results of the semantic segmentation of newspaper images. Particularly, it tries to measure how these textual features help with the variations of newspapers across time and among different newspapers.

With this goal in mind and the state of the art presented in Section 2, two hypotheses can be formulated.

Hypothesis 1: *Visual and textual features can efficiently segment images of newspapers.*

The recent progresses in both image semantic segmentation and text classification hint towards good results for the semantic segmentation of newspapers images. Indeed, the visual architecture trained for natural images have shown good adaptation to document images and that a single architecture could be adapted to different tasks [29]. Moreover, text embeddings have shown to capture representations that are able to resolve a variety of downstream tasks, from named entity recognition to question answering [35]. Using the right model and a sufficient amount of training data should give good performance on this task.

Hypothesis 2: *Visual features are less stable than textual features and will be worse at generalization.*

As discussed previously, there are strong visual and textual differences when considering different newspapers or the same newspaper across time. However, the hypothesis is that there is a high variation in term of layout, a lower variation in term of word usage and an even lower variation in term of semantic usage of discourse.

The changes in terms of layout make it quite likely that models using only visual features will fail in two different ways. The first one is when the model encounters for the first time a new layout of one of the classes it learned to recognize, an example is shown in figure 3.1. In this case, having never seen such layout, it will simply ignore it and miss it, resulting in a false negative. The second one is when an example has a very similar layout, but a different semantic meaning, as shown in figure 3.2. A purely visual model, having never been shown that this was a negative example, will likely detect it as a false positive, which will generate noise in the results.

The idea of using textual features is to tackle these two problems. Since the text is more stable, unseen visual examples could still be detected using the textual representation. Moreover, the

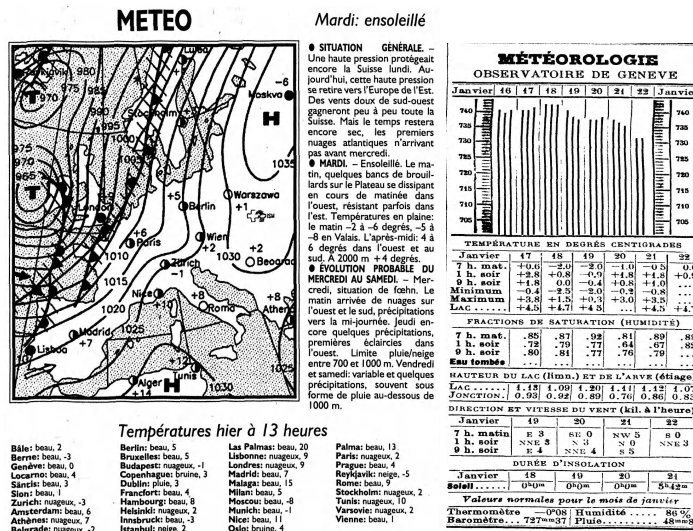


FIGURE 3.1: Two weather forecasts with distinct layouts.

Les collaborateurs et employés de l'ancienne *Maison Lugrin et Cie S.A.*, comestibles, place Longemalle, ont le regret de faire part du décès de

Monsieur Henri PELLORCE

leur cher patron.
Pour les obsèques, veuillez consulter l'avis de la famille.

NOUS DEMANDONS

Vendeur de Tapis

pour notre commerce de détail.
Ne sont priés de s'annoncer que des candidats bien recommandés, possédant connaissances approfondies de la branche tapis d'Orient et à la machine, et parlant français et allemand.
Entrée le 1er janvier ou plus tôt.
Maison de Tapis W. Geelhaar S. A., Berne, Thunstrasse 7.

FIGURE 3.2: An obituary (top) and an advertisement (bottom) with similar layouts.

number of false positives should be lower, since textual features will give contradictory signals. The underlying idea is that textual features can capture both lower level information (e.g. is this word a number?) and higher level information (e.g. is this word in the semantic field of weather?). Since the two levels of information cannot be easily learned by a purely visual neural network, textual features will thus give new information to base the prediction on.

3.2 Dataset

This section presents the data used in order to answer the different research questions. It will first generally describe the choice of newspapers, then explain the choices of classes and how they were annotated and will finally show some general statistics about the data.

Prior to this work, there were no publicly available annotated dataset of semantically annotated newspapers images. The only available datasets regarding newspapers images [46] have very broad categories, such as article, caption, header, etc. because their purposes is general article segmentation.

This project thus required an annotated dataset. It was chosen to focus on data available inside the *Impresso* project¹, in which some annotations were already available.

3.2.1 Newspapers

The newspapers chosen for the annotation are the *Journal de Genève* (JDG), the *Gazette de Lausanne* (GDL) and the *Impartial* (IMP). They are all newspapers in French, from the same geographical region (Suisse romande) and have a similar available time spans, respectively 1826-1998, 1804-1991 and 1881-2017. The digitized images are made available through the *Impresso* project.

¹<https://impresso-project.ch/>

The *Journal de Genève* and the *Gazette de Lausanne* are quite similar in term of layout, particularly towards the end of the periods, since they have been merged in 1991. The *Impartial* is more different from the two, particularly because it spans a more recent time period and has overall less serious content.

3.2.2 Classes

Newspapers contain a variety of content that changes a lot throughout time and across newspapers. This makes it difficult to find segments that are universal enough to be considered. Some general typologies of newspapers content, such as the Europeana METS ALTO Profile [47], have been defined and have served as a basis for the selection of the classes.



FIGURE 3.3: Example images for each of the classes selected. All images are from the *Journal de Genève*.

Classes selection

The classes have to match as closely as possible four criteria:

1. It has to have a visually distinct layout/appearance
2. It has to keep the semantic meaning/usage throughout time and across newspapers
3. Ideally, the layout should change throughout time and newspapers
4. Ideally, it should be source of confusion with other content items

Four classes are selected as meeting best these criteria: feuilleton, weather forecast, obituary and stock exchange table.

Feuilleton, shown in figure 3.3a, is considered as in french *roman-feuilleton* or English *serial* which is an extract of a bigger work published in several issues of a newspaper. The *Feuilleton* has a particular layout, often spanning all the columns of the newspaper and is placed at the bottom of a page. The style in which it is written is different from the rest of the newspaper, making it also particular from a textual point of view. Moreover, it can be confused with other articles that are also spanning several columns and are at the bottom of the page.

Weather forecast, shown in figure 3.3b, consist of a prediction of future weather or a report of past weather measurements. Its topic and vocabulary stays very similar semantically throughout time and across newspapers. Its layout is quite distinct from the rest of the newspaper, but it changes a lot. It can be purely textual, it can contain tables, maps or even a mix of them as shown in figure 3.1. This constant evolution makes it a good candidate to test the research question.

Obituary, shown in figure 3.3c, is a notice of the death of a person. Its layout is often characterized by thick black lines surrounding it, even though it can vary a bit across time and newspapers. The vocabulary used is very specific and stable. It can be quite easily confused with advertisements that also have thick black lines around them as shown in figure 3.2.

Stock exchange table, shown in figure 3.3d, are tables that contain the values of different stocks. They are mainly tables with numbers, which make them visually quite distinct, but also prone to confusion with other tables with numbers such as sport results and *votation* results.

Annotation process

The annotations were made using the VGG Image Annotator [48]. They were made at the pixel level and not at the instance level, meaning that if there were several elements of the same class on one page, they were not explicitly annotated as such. This was the case because there is sometimes no clear separation between two instances, for example with *Weather forecast* and *Stock exchange table*. Moreover, the segmentation in this project is also done at the pixel level. The guidelines for each classes were the following:

- *Feuilleton*, only stories published in several issues. Meaning that the *feuilleton* as the small opinion of a writer was not included. The whole *feuilleton* was included, meaning that it can span several pages.
- *Weather forecast* includes all segments linked to the forecast of weather, be it local or not. It thus includes all the different forms under which it is present, i.e. text, tables, graphs and maps.
- *Obituary* does not include the black sections around it.
- *Stock exchange table* includes only the tables, meaning that all the articles written about stock market and economy are not included.

The annotations were inevitably containing mistakes. However, most of them were corrected through iterative passes by analyzing the actual results, using the interface presented in appendix A. Particularly the false positives were closely inspected in order to see if they were not annotation mistakes. In total more than forty corrections were made using the networks predictions.

Class/Journal	JDG	GDL	IMP
Feuilleton	137	108	103
Weather	156	68	41
Obituary	153	69	102
Stocks	275	135	79
Empty	1393	697	1326

TABLE 3.1: Number of items per class.

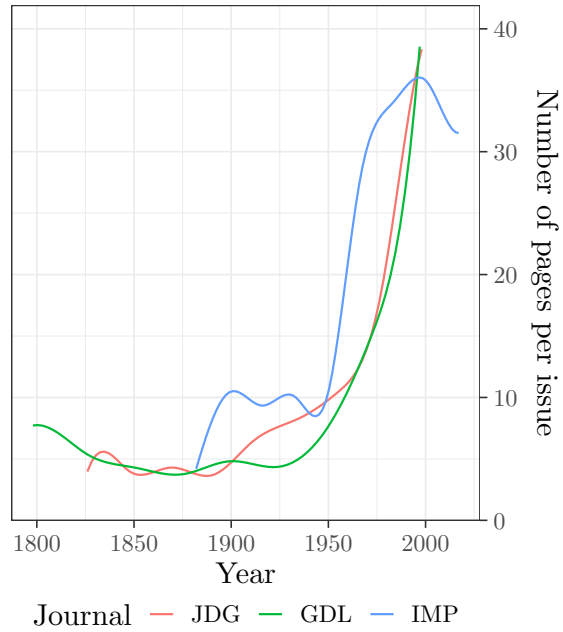


FIGURE 3.4: Evolution of the number of pages per issue through time.

3.2.3 Statistics

Annotations were made on an uniform sample for each journal. Every three years for *Journal de Genève* and every five years for *Gazette de Lausanne et Impartial*, starting from the creation of the newspaper to its end, three issues were selected. This selection resulted in 1982 pages annotated for *Journal de Genève*, 1008 for *Gazette de Lausanne* and 1634 for *Impartial*. The density of annotation is higher for *Journal de Genève* because it is the newspaper that most experiments were run on and the other two were only there to test the generalization hypothesis.

The distribution of each class can be seen in Table 3.1. The number of items represents the number of pages where at least a pixel of the class was annotated. Note that the sum of each item is not equal to the number of pages, because a page can contain several items. The vast majority of the pages are empty, meaning that they do not contain any annotation. These pages were kept during training because they are important to have a distribution as close as in a real case scenario and they help maintaining a low false positive rate by showing negative examples that may be close visually.

The Figure 3.4 shows the evolution of the number of pages for each newspaper across time. The three newspapers show a similar evolution through time with a rather constant number of pages until the 40s at which point there is an exponential augmentation.

The evolution of the number of each class of items per issue throughout time is shown on Figure 3.5. The number of items stays on average centered around one, meaning that each item appears only on one page of each issue. *Stock exchange table* however augment with time for *Journal de Genève* and *Gazette de Lausanne*, becoming more important in the newspaper and thus spanning several pages. It is also interesting to look at the evolution of the *Feuilleton* which starts to disappear in the 60s for the *Journal de Genève* and *Gazette de Lausanne*, but stays constant for *Impartial*.

This evolution means that time-wise, there will be less *Feuilleton* from recent years and less *Stock exchange table* from older years. Across newspaper the difference will mainly be *Journal de*

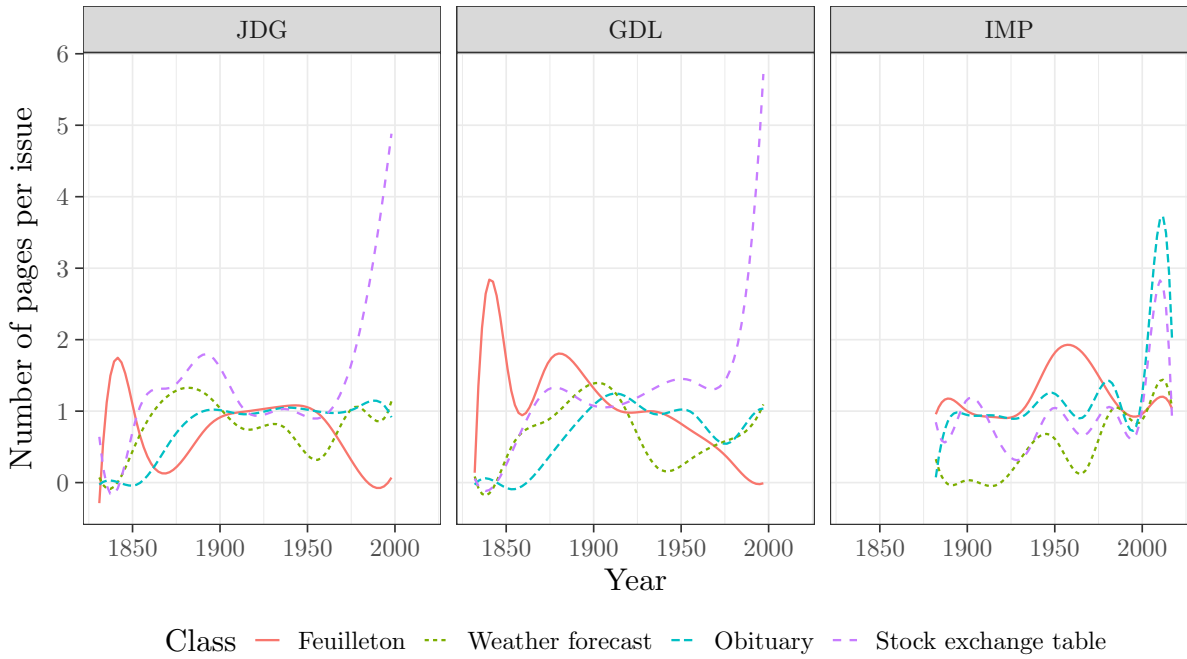


FIGURE 3.5: Evolution of the number of items of each classes per issue per journal through time.

Genève and *Gazette de Lausanne* versus *Impartial*, where the latter has data from after 2000, more obituaries and *Feuilleton* and less *Weather forecast*, whereas the former will stop before 2000 and will have more stock exchange tables.

3.3 Text embeddings map implementation

This project mainly uses the idea introduced in [45], meaning that it maps text embeddings into a 2D space. The building of the mapping will be referred to as "text embedding map". The mapping is given an image of size $H \times W$ and a list of tokens T present in the image. These tokens will be mapped to a text embedding map G of size $H \times W \times N$, where N is the dimension of the embeddings. The tokens are obtained through an OCR process and can either be characters, words or even lines. To each token t is associated a bounding box which is also a product of the OCR process. All the pixels contained in the bounding box of the token t are defined as the set $\mathbf{b}_t \in \mathbb{R}^2$. Then each pixel $g_{i,j}$ of the text embedding map is computed with

$$g_{ij} = \begin{cases} E(t) & \text{if } (i, j) \in \mathbf{b}_t \\ E(0) & \text{otherwise} \end{cases}$$

where $E(t)$ is a mapping of $t \rightarrow \mathbb{R}^N$ which corresponds for example to a word embedding and $E(0)$ correspond to a null token that is then usually mapped to the vector 0^N . Thus each pixel that overlaps with a bounding box of a token is mapped to its corresponding embedding. If a pixel overlaps with two bounding boxes, it is attached to the one that has the closest center.

The final result is a text embedding map which is a 3D matrix with the two first dimensions being the document image representation of the text and the last dimension being the embedding dimension. This can directly be used in the dhSegment architecture, as it has the same shape as the results of convolutional layers.

3.4 Model architecture

The model architecture is described in Figure 3.6. It is a modified version of dhSegment [29] that uses an encoder-decoder feature as first introduced in [1] with a Resnet-50 [30] as an encoder (left side of the image). The encoder is pre-trained on ImageNet and is slightly modified to limit the number of feature channels to 512. Each step of the decoder part (right side of the image) upsamples by two the lower level features, concatenates it with the corresponding higher dimension feature of the encoder and passes the two in a 3x3 convolution to halve the number of features. The feature maps thus become larger and thinner when they go “up” the decoder and the dimensions mirror the ones of the encoder. Finally the last layer is a 1x1 convolution to reduce the number of features to the number of classes to predict and its output goes into a softmax function to get the probabilities.

The modified architecture adds a text embedding map that is concatenated to the feature channels. In Figure 3.6 the three possible positions of the concatenations are shown with (1), (2) and (3). Note that the dimension of the embeddings map shown in the figure is of 300, this can of course vary depending on the embedding.

3.4.1 Concatenation of the embeddings map

The text embeddings map can be input at different levels of the network, each of the levels corresponding to different assumptions. By inputting the map at the beginning of the network, it is assumed that it can be treated as an image, i.e. that convolving on a 2D map of textual embeddings is meaningful. By inputting the map at the end, the map will rather give additional information than just the image features, while making less assumption on the 2D properties of textual embeddings.

3.4.2 Size and nature of the embeddings

The embeddings themselves are a parameter that can be adjusted. Indeed, as presented in section 2.3 there are several ways to generate embeddings from text. It is possible to have embeddings that are context dependent or independent, embeddings that are made at the word or character level and finally reduce the size of the original embeddings.

Type of embeddings

Three types of embeddings are considered in this project.

Fasttext embeddings are context-independent, meaning that each word has a unique embedding regardless of its context and are computed at the word level. The implementation used is the one of the flair² library where the french model is computed on a french Wikipedia dump and outputs embeddings of size 300. This library does not make use of the subword feature of Fasttext. All the words that are outside of the vocabulary of the french Wikipedia dump or that contain OCR mistakes are mapped to an all zero vector. Approximately 25% of the words in the corpus were mapped to zero. These embeddings will allow to show how non-contextual embeddings perform. Fastest embeddings are more stable because they do not depend on their context, which could make it easier in some cases for the network to pick interesting features.

²<https://github.com/zalandoresearch>

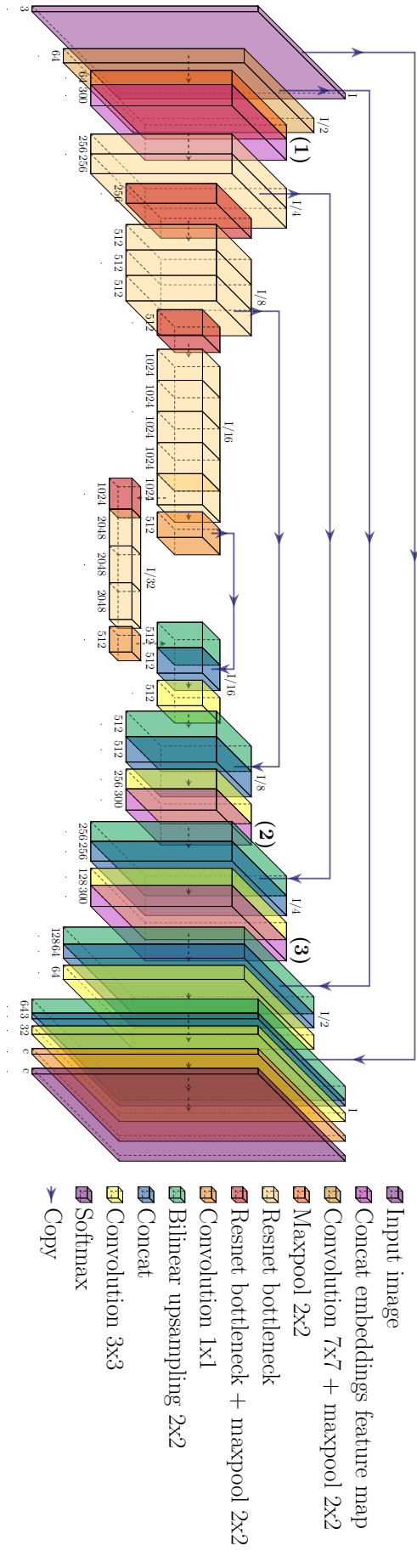


FIGURE 3.6: The model architecture used.

The (1), (2), (3) markers show where the textual feature maps are concatenated to the visual feature map. Note that in the actual network, the concatenation is done only at one of the three positions.

The size relative to the original image size I is indicated at each step of the network. Moreover the dimension of the features maps at each step is indicated below the blocks, considering that an embeddings feature map of size 300 has been input at (1).

Flair embeddings are context-dependent, meaning that each word will have a different embedding depending on its context, and are computed at the character level. Once again the implementation used is the one of the flair library where the model is trained on a french Wikipedia dump with a character window size of 250 and outputs embeddings of size 2048. The model used is a stacked embedding between a forward and backward model each of size 1024. The fact that the embedding is done at the character level makes it more resilient to out of vocabulary words and OCR errors, having less than 1% of missed words. The ability of those embeddings to capture the context of the word could help in discriminating higher level features of the text.

Stacked embeddings are the stacking of the two previously presented embeddings. As proposed in [49], it is possible to simply stack the embeddings and then use them for a downstream task, letting the neural network choose which representation is the best for the task. This approach thus gives embeddings of size $2048 + 300 = 2348$.

Size of the embeddings

This size of the embeddings can quickly become a bottleneck in the model for both computational resources and in term of what can be learned. Indeed, the images have an input size of $5 \cdot 10^5$ pixels, and since each pixel is mapped to an embedding vector, the memory size rapidly explodes. It is thus not possible to input the embeddings feature map directly at the highest resolution level in the network. Moreover, the number of features needed to discriminate between the classes presented in section 3.2.2 should be relatively small.

These limitations thus encourage to find ways to reduce the dimension of embeddings, be it through a general dimension reduction method or through learning a representation in an end-to-end manner inside the model architecture.

Principal component analysis

Principal component analysis (PCA) is one of the most commonly used dimensionality reduction technique. It finds an uncorrelated linear mapping that maximizes the variance in the lower dimension.

It is computed by taking all the word vectors for a particular embedding type and stack them into a matrix M of dimension $V \times N$ where V is the vocabulary size and N the embeddings size. Then the transformation matrix T is computed with the singular value decomposition of M . In order to reduce a vector x of size N to a new vector \hat{x} of size M , it suffices to compute $\hat{x} = x \cdot T_M$ where T_M is a truncated matrix with only the first M principal components and is thus of size $N \times M$.

The figure 3.7 shows the potential of both word embeddings and PCA. Even in three dimension, PCA is able to keep important information about the text. For example the numbers are shown in grey, the punctuation in green and the stop-words are more yellowish.

End-to-end representation learning

Another way to reduce the dimension of the vector is to learn a smaller representation for the embeddings directly inside the model.

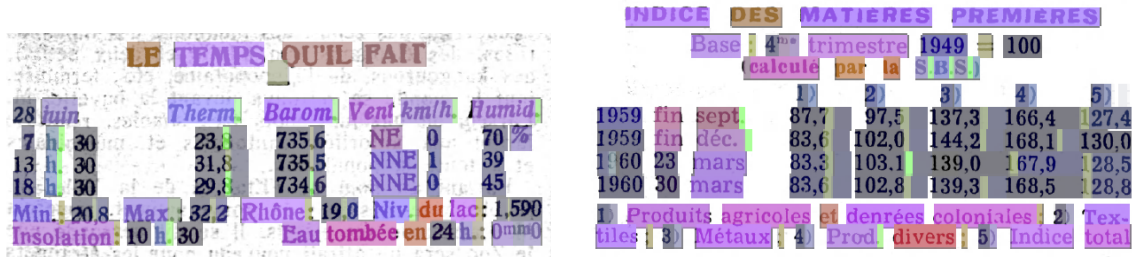


FIGURE 3.7: Projection of an embedding map with Flair vectors on three dimension (red, green and blue) using PCA. On the left is an example of a weather forecast and on the right a stock exchange table. It can be seen that they both have very similar layout, with a title, a table with a similar number of columns and some text under it. However, even in three dimension (down from the 2048 dimension of the original vector), it is possible to see some differences between the two that could not be easily caught by visual features. Such that the weather forecast has a column with letters only.

In this project, it was chosen that the dimensionality reduction would be learned in an end-to-end manner. Several projections matrices T_k, \dots, T_k are learned to go from the original dimension to the desired dimension. A projection matrix, in order to project from a dimension N to a dimension M , is simply a stack of M 1D convolutions with a kernel size of one. These convolutions can be thought of as matrices of sizes $N \times M$ and operate on one embedding at a time as does PCA.

The number of projection matrices computed depends on the input size of the vector and the desired final dimension. The size of the first dimension reduction is of 256, 1024 and 1024 respectively for Fasttext, Flair and the stacked embeddings. These numbers were chosen to be powers of two and still reduce the dimension of the embeddings. The next projections are always half of the previous projection until the smallest power of two larger than the target dimension is reached. The final projection is of the size of the target dimension. For example, with an input size of 2348 and a target dimension of 256, three projection matrices are computed, respectively of sizes 1024, 512 and 256.

3.5 Evaluation methodology

This sections presents the different metrics used to compare and evaluate the experiments. The intersection-over-union will be used as general metric for comparing different experiments. Precision and recall will be used to assess the performances of the models with more real-world scenarios.

3.5.1 Mean Intersection over Union

The intersection-over-union (IoU) is one of the most standard metric for semantic segmentation. It measures how well two sets are aligned. Figure 3.8 gives a quantified example. It is important to note that this metric is measured at the image level and not at the instance level, because the annotations are made at the pixel level and thus contain no information on the instances.

It is computed as follows. Let i be an image belonging to the set of images I . Let c be a class belonging to the set of classes C . Let P_{ic} be the set of pixels predicted for image i belonging to



FIGURE 3.8: Example of the behaviour of the IoU metric. Figure 3.8a shows the pixels of the manual annotation (ground-truth). Figure 3.8b shows a bad prediction and thus the IoU is very low. Figure 3.8c shows a good prediction and thus the IoU is very high. The actual threshold at which an IoU is considered as good strongly depends on the practical use case.

class c and G_{ic} be the set of ground-truth pixels of the annotation of image i belonging to class c . Then the IoU for the image i and class c is:

$$IoU_{ic} = \frac{|P_{ic} \cap G_{ic}|}{|P_{ic} \cup G_{ic}|}$$

The mean intersection-over-union (mIoU) for a class c is then the average of the IoU of the image $i \in J = \{i \in I \mid |P_{ic} \cup G_{ic}| > 0\}$ i.e. where either the predicted or the ground-truth set have at least a pixel of class c (the true negatives are not counted), then the mIoU is:

$$mIoU_c = \frac{1}{|J|} \sum_{j \in J} IoU_{jc}$$

3.5.2 Precision and Recall

The intersection over union does not directly give information on the performance regarding true positive (TP), true negatives (TN), false positives (FP) and false negatives (FN). It was thus

important to have metrics to assess those as they are useful measures when considering more practical use cases.

What is usually done in segmentation is to consider an example as positive when it reaches a particular threshold $\tau \in [0, 1]$ of IoU. Then a prediction with an IoU $\geq \tau$ is considered as a TP, a prediction with no IoU is a TN, a prediction with an IoU of zero and no predicted pixels is a FN and the rest are considered as FP.

This thus gives the possibility to compute different measures at a threshold τ :

$$\text{Precision at } \tau = P@_{\tau} = \frac{TP}{TP + FP}$$

$$\text{Recall at } \tau = R@_{\tau} = \frac{TP}{TP + FN}$$

The precision is useful to measure the noise that the approach generates. It computes the percentage of items that were rightly classified (TP) over all detected items (TP+FP). The recall is for computing how much is missed by the approach. It computes the percentage of rightly detected items (TP) over all the items belonging to the class (TP+FN).

3.6 Model variance

Neural networks are inherently random. Their weights are initialized randomly and are then optimized using gradient descent. This randomness can cause the model to have a large variance in term of results. This means that the exact same model trained twice for the same amount of steps with the same data can have different results. This is due to the existence of numerous local minima where the optimization algorithm can get stuck without ever finding the global minimum. This is more likely to occur when the data is scarce and complex. This is the case in this project since each classes has less than 100 samples in the training set and there are a lot of differences inside the classes because of the evolution through time. It was thus necessary to find ways to reduce the variance and a methodological framework in order to assess the quality of the results even with large variance.

Reducing variance

Several steps were taken in order to reduce the variance, mainly by fixing the randomness as much as possible. The training and testing sets were fixed, as well as the order of appearance of the training data for every models. This last step is very important as this ensure that every network sees the same example at the same time, thus encountering the same difficulties. It was however not relevant to fix the weights initialization of the network. Indeed, the architecture differs from one model to another, plus the performances would then be mainly tied to luck, since the weights could be bad for an architecture and good for another.

Interpreting results

The first step to be able to interpret results is to have a good estimation of the variance in order to ensure that the results are not obtained by pure luck, as shown in [50]. In this study, the authors trained the exact same model several times and separated the results in two groups,

pretending that these were two different models. They showed that when reporting only the result of one training or the best result of several trainings, it was not possible to distinguish the two identical models in between 10% and 30% of the cases. Even worse, the difference between the two models was found to be statistically significant. The authors thus argue the necessity to report the mean and standard deviation of several runs. Moreover, they propose to use Welch's t -test in order to find out if the difference of means between two approaches is significant. This test is more reliable than Student's t -test when the two samples have unequal variance and is more robust to false positive [51]. However, this test requires that the two samples come from a normal distribution.

In this project, each model is trained ten times and the mean and standard deviation of its performances is reported. Moreover, the normality of the samples is assessed using quantile-quantile plot. If the distributions is assumed to be normal, Welch's t -tests are performed to compare the approaches. Each time a result is reported as significant in Section 4, it means that its p -value for the test is less than 0.05.

4 Experiments

This section introduces and discusses the different experiments of this project. Section 4.1 explains the experiment setup that was used in order to perform the experiment. Section 4.2 introduces the baseline model that does not use textual features. Section 4.3 presents and discusses the different experiments that try to optimize the performances over the baseline. Finally, section 4.4 presents and discusses the experiments that explore the generalization of the different approaches.

4.1 Experimental setup

Training setup

The models were trained with the default weight decay regularization of dhSegment, i.e. 10^{-6} . The input of the network was the entire images resized to have $5 \cdot 10^5$ pixels. As explained in section 3.6 the order of the input was the same for every training. The images were also randomly augmented on-the-fly by scaling ($s \in [0.8, 1.2]$) and rotation ($r \in [-0.01, 0.01]rad$). The weights were randomly initialized using Xavier initialization [52]. The optimizer used is Adam [53] with an exponentially decaying learning rate of 0.95 starting at 10^{-4} . All the networks were trained with a batch size of four and using batch renormalization [54]. The batch size was chosen to be as big as possible as long as the biggest network was able to fit on the GPU used in order to gain something from renormalization. Each network was trained for 50 epochs, resulting in around 17'000 steps.

Training data

Class	Train count	Test count	Train ratio (%)	Test ratio (%)
Feuilleton	101	36	7.28	6.05
Weather forecast	103	53	7.43	8.91
Obituary	107	46	7.71	7.73
Stock exchange table	189	86	13.63	14.45
Empty	982	411	70.80	69.08

TABLE 4.1: Distribution of the classes for the training and testing sets.

Only *Journal de Genève* was splitted into training and testing sets, the other newspapers being only used for testing purposes. The splitting ratio was of 70% training and 30% testing. In total there were 1387 images used for the training set 416 for the test set. The actual distribution of each classes can be found in table 4.1. It can be noted that the ratio of the classes is very well balanced between the training and testing sets.

The images are obtained in the JPEG2000 format and are down-scaled to an height of 2000 pixels in PNG format. The OCR of the images is obtained on the JPEG2000 using *Transkribus*¹ which itself uses *ABBYY FineReader 11*². The performances of the OCR was not strictly assessed, however Fasttext embeddings had 25% of missing words.

Post-processing

All the results reported here are obtained with minimal post-processing. Given the final probability map, the largest probability larger than 50% was selected as the final class. Then in order to avoid too small predictions, the connected components that had an area smaller than .5% of the total image area were discarded.

Material/Software

The networks were trained on Kubernetes clusters with a NVIDIA Tesla V100 GPU with 32GB of memory. The models were created using the Tensorflow³ library version 1.13.1. The code of the modified architecture can be found online⁴.

Metrics choice

The mIoU is the most commonly reported metric in image segmentation. Indeed, it provides a good summary of the performances and has no parameters unlike the other metrics presented. It will thus be the metric used in the experiments when comparing different competing approaches.

The other metrics considered are the precision and the recall. These metrics better describe the real-world performances of the models. The precision gives information on the number of false positive. It will thus measure how much noise the approach is generating. The recall gives information on the number of false negative. Hence, it measures how many results are missed. It is important to note that this metric is biased by its construction. Indeed, all the examples found with an insufficient threshold are counted as false positive and thus are not counted in this metric. Only the number of true positives change with the threshold, whereas the number of false negative stay constant. The recall gives only information about the percentage of examples that were fully missed and not the one partially missed.

Finally, two different thresholds are considered for the precision and recall measures. A threshold of 60 can be considered as the lowest threshold of interest for most downstream tasks. It is good enough for *Feuilleton* for example, since it would be easy to retrieve the missing part only from the information that there is a *Feuilleton* on the page. However, for the other classes that are not necessarily made of one continuous block, this threshold is not always sufficient. This is why the threshold of 80 is also considered. It is obviously much harder to reach and the results will always be lower than with a threshold of 60. The difference of performance between the two thresholds will show how robust the model is.

¹<https://transkribus.eu>

²<https://www.abbyy.com/en-us/support/finereader-11/>

³<https://www.tensorflow.org/>

⁴<https://github.com/raphaelBarman/dhSegment>

Class	mIoU	P@60	P@80	R@60	R@80
Feuilleton	74.12 ± 7.59	82.02 ± 7.24	66.45 ± 14.87	97.91 ± 1.95	97.52 ± 2.26
Weather	81.27 ± 2.18	91.08 ± 4.93	66.94 ± 7.41	78.74 ± 1.98	73.06 ± 2.57
Obituary	75.37 ± 2.98	83.37 ± 4.46	67.37 ± 2.04	93.02 ± 2.88	91.49 ± 3.67
Stocks	83.11 ± 0.88	89.21 ± 1.42	80.51 ± 2.19	93.78 ± 0.73	93.15 ± 0.81
Average	79.3 ± 2.29	86.86 ± 2.6	72.29 ± 3.74	90.64 ± 1.03	88.94 ± 1.4

TABLE 4.2: Results for the vanilla model.

4.2 Baseline model

The baseline model will be dhSegment without textual feature map added. This model will be referred to as Vanilla. The results for this baseline are presented in Table 4.2.

The standard deviation is quite high for some metrics. This is particularly true for the precision measure. It can be explained because some models have more results at the limit of the threshold, thus these results will be counted as true or false positive depending on which side of the threshold they are.

For the precision at 60, the results are relatively good for *Weather* and *Stocks* with only one out of ten false positive. The results are a bit worse for *Feuilleton* and *Obituary*. This is expected, since *Weather* and *Stocks* are the two classes that are the most visually distinct. Whereas *Feuilleton* and *Obituary* are more text based.

For the precision at 80, we notice a big drop for almost every classes except for *Stocks*. The drop is particularly strong for *Weather* which loses 25% in precision. This shows that the network does not make too many mistakes when finding the items, but has trouble finding their precise location. When considering this threshold, which may be necessary for some downstream tasks, it becomes clear that the network is not performing well enough. Apart from stocks, more than 35% of the results are false positives, which makes them too noisy for many applications.

The recall is overall higher than the precision and the difference between the threshold is less pronounced, mainly because of the bias explained in the Section 4.1. Its results are satisfying for *Obituary* and *Stocks*, where around 7% are fully missed and very satisfying for *Feuilleton*, where only 3% are fully missed. The recall is worse for *Weather*, indicating that a lot of examples of this class are completely missed. This result is certainly due to the fact that some examples of *Weather* are purely textual and are thus more difficult to identify.

4.3 Embeddings experiments

In this section, the experiments will provide an exploration of the different type of embeddings, input level and dimensionality reductions techniques. The goal is to see how these different parameters influence the results and how their scores compare to the baseline model.

4.3.1 Embeddings type and input level

Experiment description

The objective of the first experiment is to determine how the type of embeddings and the input level influences the results.

There are three possible types of embeddings that are considered:

- Non contextual embeddings, which will be Fasttext because of its performances and ability to encode to the subword level.
- Contextual embeddings, which will be Flair because it is computed at the character level which makes it more resilient to out of vocabulary words and OCR errors.
- The stack of the two previous embeddings ("Flair-FT").

The embeddings have respectively for dimension 300 ("D-300"), 2048 ("D-2048") and 2348 ("D-2348").

In this experiment no pre-processing is applied to textual feature maps. This setup is thus the identity pre-processing and it will be referred as "ID" in the experiments. Not reducing the dimensions of the embeddings in a pre-processing step may result in memory heavy maps depending on the embeddings and its input resolution and has to be taken into account when choosing the input level.

The second parameter is the input level inside the network for the embeddings feature map. Indeed, as explained in Section 3.4.1, the place of input makes different assumptions. If the map is input at the beginning, it will pass through most of the network and will thus be treated more equally like a visual feature. Whereas if it is input at the end of the network, it will rather add some additional contextual information to the image feature map to make the final decision of the pixel class. The exact place of the input is shown in more details in Figure 3.6. The idea was to input the text embeddings at the highest resolution possible in the network, while having enough GPU memory to keep a batch size of four. When input in the encoder, all the embeddings were placed at position (1) (named position "Enc-0", in the Figure 3.6) and at position (3) (named "Dec-1") in the decoder except for the stacked embeddings that were input at (2) (named "Dec-2"). Indeed, as explained in the previous paragraph, the different size and thus memory usage of some embeddings did not allow them to be input at the same level as the others.

To sum up, six variations are analyzed:

1. **Fasttext ID Enc-0 D-300**, Fasttext embeddings, not pre-processed, input at the beginning with a dimension of 300
2. **Fasttext ID Dec-1 D-300**, Fasttext embeddings, not pre-processed, input at the end with a dimension of 300
3. **Flair ID Enc-0 D-2048**, Flair embeddings, not pre-processed, input at the beginning with a dimension of 2048
4. **Flair ID Dec-1 D-2048**, Flair embeddings, not pre-processed, input at the end with a dimension of 2048
5. **Flair-FT ID Enc-0 D-2348**, Stacked embeddings, not pre-processed, input at the beginning with a dimension of 2348

6. **Flair-FT ID Dec-2 D-2348**, Stacked embeddings, not pre-processed, input at the end with a dimension of 2348

Results and discussion

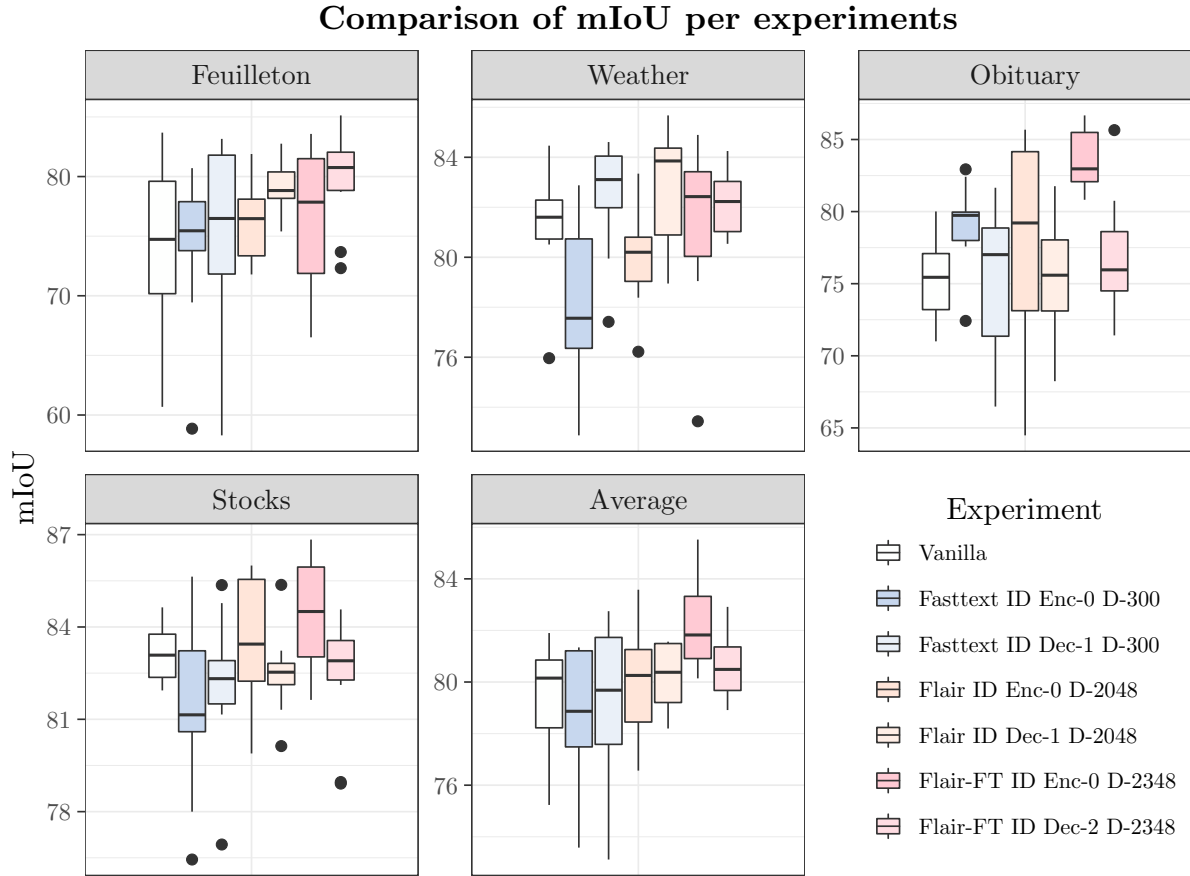


FIGURE 4.1: Box plot of the results with type of embeddings and input location variations.

Experiment / Class	Feuilleton	Weather	Obituary	Stocks	Average
Vanilla	74.12 \pm 7.59	81.27 \pm 2.18	75.37 \pm 2.98	83.11 \pm 0.88	79.29 \pm 3.18
Fasttext ID Enc-0 D-300	74.44 \pm 6.42	78.25 \pm 3.14	79.09 \pm 2.93	81.39 \pm 2.79	78.83 \pm 2.55
Fasttext ID Dec-1 D-300	75.35 \pm 7.99	82.46 \pm 2.27	75.32 \pm 5.04	82.18 \pm 2.31	80.24 \pm 1.26
Flair ID Enc-0 D-2048	76.08 \pm 3.27	80.01 \pm 1.96	77.86 \pm 7.1	83.46 \pm 2.27	80.03 \pm 2.26
Flair ID Dec-1 D-2048	79.12 \pm 2.36	82.88 \pm 2.26	75.12 \pm 4.35	82.5 \pm 1.35	80.58 \pm 1.35
Flair-FT ID Enc-0 D-2348	76.73 \pm 5.9	81.38 \pm 3.34	83.58 \pm 2.02	84.43 \pm 1.84	82.16 \pm 1.72
Flair-FT ID Dec-2 D-2348	79.85 \pm 4.14	82.16 \pm 1.24	76.86 \pm 4.08	82.4 \pm 1.95	79.3 \pm 2.29

TABLE 4.3: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %).

Results for the mIoU metric are presented in Table 4.3 and in Figure 4.1 under the form of box-plots. Box-plots are used to get a better understanding of the variance of the different models and as visual way to see the difference between them. As explained in section 3.6, it can be seen that the standard deviation is quite high for all models. It is even more striking when looking at the box plots. Hence, the analysis is made considering only the mean, but it is emphasized

when the difference is statistically different as explained in section 3.6. This significance testing is done using Welch's t -test with a p -value < 0.05 .

Overall, the average mIoU of all classes is consistently higher than vanilla for every embedding, between .8% and 2.9%. This hints that there is gain to add a textual feature map to the architecture. This difference is significant (p -value < 0.05) for the stacked embeddings input at the beginning of the network ("Flair-FT ID Enc-0 D-2348"), where its mean is statistically higher than every other method. Compared to Vanilla, its mean is higher between .9% and 4.8% at a 95% confidence interval. This indicates that the stacking of the two embeddings has a better representation power, which is consistent with the reported results of the Flair paper [36]. Moreover, the fact that it performs better when input at the beginning of the network would indicate that the network manages to learn in an end-to-end manner a new representation better suited for the segmentation task.

For *Feuilleton*, there is no significant difference between the vanilla model and the other ones, this is mainly due to the large variance. However, the two models that use Flair embeddings input in the decoder have higher mean and less variance.

Flair embeddings performing better than Fasttext embeddings could be explained by the fact that the textual composition of *Feuilleton* is contextual. Indeed, *Feuilleton* is characterized by a narration voice that could only be picked up by contextual embeddings. Another tendency is that embeddings maps input at the end are always a bit better than when input at the beginning. This effect is even significant between the two Flair variants, where the one with input at the end as a higher mean between 0.34% and 5.7% at a 95% confidence interval. This could be due to the fact that there is no real gain to fine-tune the representation, either because of over-fitting either because the embeddings without transformation are already good enough for discriminating this class.

For *Weather*, there is again no significant gain with textual feature maps. The only noticeable tendency that emerges is that, similarly to *feuilleton*, the map input at the end are better. This effect is significant between the couple of Fasttext embeddings with a gain at 95% confidence interval between 1.6% and 6.8% and the couple of Flair embeddings with a gain between 0.9% and 4.9%. This again could be explained fact that embeddings do not need to be processed by the whole network in order to detect *Weather* class. Moreover, since the result is quite similar for Fasttext, Flair and the stacked embeddings when input at the end, it indicates that there is no real benefit of having context, but rather that the words used in this class are distinct enough.

For *Obituary*, both "Fasttext ID Enc-0 D-300" and "Flair-FT ID Enc-0 D-2348" have means that are significantly higher than "Vanilla", respectively a gain with 95% confidence interval of between 0.94% and 6.5% and of between 5.8% and 10.6%. The gain over vanilla can be explained through the relatively high number of false positive that the purely visual approaches generates. Indeed, there is a lot of advertisement in the newspapers that are easily mistaken, as shown in Figure 3.2. However, when looking at textual features only, obituaries are very distinct. They have a very specific and small vocabulary that is easily picked by most of the models using textual features.

For *Stock exchange table* there is no significant difference between vanilla and other approaches. This could be explained by the fact that stocks is a class that has very particular visual features. There is of course other tables with numbers that are confused with this class, but overall there is very few false positives.

Experiment		Vanilla	Flair ID Enc-0 D-2048	Flair ID Dec-1 D-2048	Flair-FT ID Enc-0 D-2348	Flair-FT ID Dec-2 D-2348
Class	Metric					
Feuilleton	mIoU	74.12 \pm 7.59	76.08 \pm 3.27	79.12 \pm 2.36	76.73 \pm 5.9	79.85 \pm 4.14
	P@60	82.02 \pm 7.24	84.45 \pm 3.4	84.79 \pm 2.22	83.24 \pm 7.17	84.16 \pm 4.1
	P@80	66.45 \pm 14.87	75.31 \pm 9.02	79.66 \pm 4.07	71.54 \pm 15.05	81.19 \pm 5.11
	R@60	97.91 \pm 1.95	100.0 \pm 0.0	98.24 \pm 1.51	100.0 \pm 0.0	98.26 \pm 1.5
	R@80	97.52 \pm 2.26	100.0 \pm 0.0	98.12 \pm 1.63	100.0 \pm 0.0	98.21 \pm 1.54
Weather	mIoU	81.27 \pm 2.18	80.01 \pm 1.96	82.88 \pm 2.26	81.38 \pm 3.34	82.16 \pm 1.24
	P@60	91.08 \pm 4.93	91.1 \pm 3.74	92.46 \pm 4.67	91.81 \pm 4.67	91.18 \pm 2.53
	P@80	66.94 \pm 7.41	70.08 \pm 5.23	71.81 \pm 7.56	71.37 \pm 7.28	72.47 \pm 2.69
	R@60	78.74 \pm 1.98	82.36 \pm 1.47	80.58 \pm 1.68	87.14 \pm 2.75	80.53 \pm 1.55
	R@80	73.06 \pm 2.57	78.09 \pm 3.25	76.22 \pm 2.74	84.0 \pm 3.32	76.64 \pm 2.38
Obituary	mIoU	75.37 \pm 2.98	77.86 \pm 7.1	75.12 \pm 4.35	83.58 \pm 2.02	76.86 \pm 4.08
	P@60	83.37 \pm 4.46	84.5 \pm 8.89	83.43 \pm 5.36	91.27 \pm 2.8	85.05 \pm 4.52
	P@80	67.37 \pm 2.04	70.4 \pm 7.54	67.25 \pm 5.15	80.89 \pm 3.88	70.49 \pm 6.59
	R@60	93.02 \pm 2.88	94.15 \pm 2.97	95.41 \pm 2.42	90.37 \pm 1.44	94.93 \pm 2.58
	R@80	91.49 \pm 3.67	92.98 \pm 3.87	94.35 \pm 2.93	89.27 \pm 1.48	93.98 \pm 2.95
Stocks	mIoU	83.11 \pm 0.88	83.46 \pm 2.27	82.5 \pm 1.35	84.43 \pm 1.84	82.4 \pm 1.95
	P@60	89.21 \pm 1.42	89.09 \pm 1.88	89.38 \pm 1.66	90.11 \pm 1.77	88.86 \pm 2.19
	P@80	80.51 \pm 2.19	80.44 \pm 3.44	81.26 \pm 1.85	83.49 \pm 2.11	80.01 \pm 2.67
	R@60	93.78 \pm 0.73	93.88 \pm 0.9	94.5 \pm 0.86	93.73 \pm 1.08	93.97 \pm 0.7
	R@80	93.15 \pm 0.81	93.26 \pm 1.04	93.98 \pm 0.96	93.27 \pm 1.14	93.35 \pm 0.74
Average	mIoU	79.3 \pm 2.29	80.03 \pm 2.26	80.24 \pm 1.26	82.16 \pm 1.72	80.58 \pm 1.35
	P@60	86.86 \pm 2.6	87.47 \pm 2.01	87.75 \pm 1.34	89.43 \pm 1.79	87.55 \pm 1.53
	P@80	72.29 \pm 3.74	75.09 \pm 4.1	75.86 \pm 1.67	78.07 \pm 3.76	76.53 \pm 2.17
	R@60	90.64 \pm 1.03	92.12 \pm 0.9	91.97 \pm 0.75	92.39 \pm 0.95	91.65 \pm 0.62
	R@80	88.94 \pm 1.4	90.92 \pm 1.18	90.82 \pm 0.91	91.37 \pm 1.0	90.55 \pm 0.78

TABLE 4.4: Results of the different metrics for the embeddings and input location experiment.

Other metrics

This section explores how the other metrics behave with the different models. It was chosen to not report Fasttext results as they were below the other embeddings types. The metrics are reported in Table 4.4, for each experiment and class, its mIoU is again reported for comparison purposes.

Overall, the results are similar than with the mIoU metrics. The models using textual features performing better than vanilla. “Flair-FT ID Enc-0 D-2348” gaining versus Vanilla 3% and 6% for precision at threshold 60 and 80 and around 2% for the recall. The text-based model show also more robustness, by having a smaller drop than vanilla when comparing the 60 and 80 thresholds. For downstream tasks, the results are however not different enough to make a real difference.

For Feuilleton, the results are also similar than when considering only the mIoU. However, “Flair ID Enc-0 D-2348” also has a high precision at 60 threshold. When comparing to vanilla, the gain of textual features is particularly marked when considering the precision at 80 threshold. The models lose very little precision compared to a lower threshold and still maintain a relatively high precision.

The two models taking into input the embeddings at the beginning both reach perfect recall scores at both threshold, meaning that no example of *Feuilleton* is completely missed. These two models, while generating noise with false positive, are thus interesting to consider when the recall is important.

For *Weather* the results are similar than with *Feuilleton*, with more robustness in the precision between the two thresholds. The recall is significantly improved with “Flair-FT ID Enc-0 D-2348” versus Vanilla. This means that the network with textual features is able to retrieve more examples, certainly text-based, of this class.

For *Obituary*, the precision results are significantly better with textual features, particularly with “Flair-FT ID Enc-0 D-2348”. The network reaches more than 90% precision at a threshold of 60 and still manages to reach 80% at a threshold of 80. This kind of score begins to be good enough to consider to use the network for downstream tasks.

For the recall, there is no significant differences between the networks, except for “Flair-FT ID Enc-0 D-2348” which is worse than the other networks. This means that this network is more conservative in its predictions, thus it maintains a lower false positive rate, but a higher false negative one.

For *Stocks*, there is no noticeable difference in term of precision and recall between all the models.

Experiment summary

There is a gain to add textual information even not processed inside a segmentation network that usually works only at a visual level. The gain is particularly strong with *Feuilleton* and *Obituary* where the performances are significantly higher. This result is interesting because these two classes are the most textual ones among the four, meaning that the network is able to pick the textual information that characterize them. The precision of 91% reached for *Obituary* at threshold 60 could be good enough depending on the use case. Moreover the perfect recall achieved for *Feuilleton* with some models is encouraging if the goal is to find all of them, since it would just suffice to filter the false positives in a second pass.

4.3.2 Blank embeddings

Experiment description

This experiment tries to determine the representation power of the embeddings. The images are thus replaced by blank version of them, i.e. input vectors of the same size but all filled with zeros. Hence, the network can only make its prediction based on the textual embeddings maps.

Since the network still needs to learn something about the images, only the variant of the models where the map is input at the beginning of the encoder is considered. Indeed, when input at the end of the decoder, the network does not manage to learn anything since the rest of the network only sees blank images.

Results and discussion

Results for the mIoU metric are presented in Table 4.5 and in Figure 4.2.

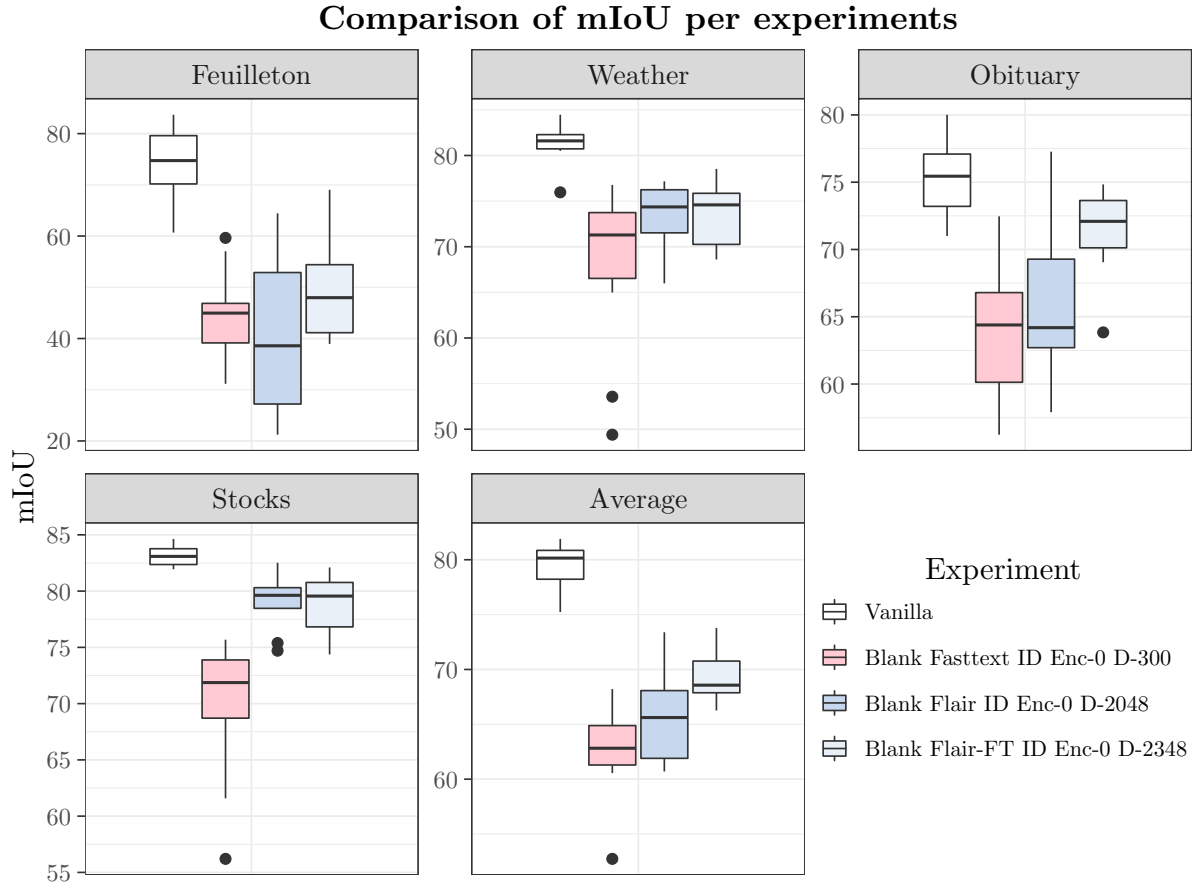


FIGURE 4.2: Box plot of the results with different dimensionality reduction. “Flair-FT ID Enc-0 D-2348” is reported again for comparison purposes.

Overall the results are good considering that only the embeddings maps are used and the network was not optimized for that. It even reaches competitive results when compared to Vanilla for *Obituary* and *Weather*, losing only 4%.

The stacked variant of embeddings is consistently higher and the ranking between the embeddings is also consistent:

1. Stacked embeddings
2. Flair embeddings
3. Fasttext embeddings

Experiment / Class	Feuilleton	Weather	Obituary	Stocks	Average
Vanilla	74.12 ± 7.59	81.27 ± 2.18	75.37 ± 2.98	83.11 ± 0.88	79.3 ± 2.29
Blank Fasttext ID Enc-0 D-300	44.6 ± 8.87	67.98 ± 9.26	63.89 ± 5.48	69.75 ± 6.22	62.47 ± 4.14
Blank Flair ID Enc-0 D-2048	40.2 ± 15.57	73.35 ± 3.89	66.25 ± 6.75	79.08 ± 2.46	65.92 ± 4.7
Blank Flair-FT ID Enc-0 D-2348	49.05 ± 9.41	73.55 ± 3.55	71.44 ± 3.26	78.87 ± 2.61	69.3 ± 2.25

TABLE 4.5: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %).

Experiment		Vanilla	Blank Fasttext ID Enc-0 D-300	Blank Flair ID Enc-0 D-2048	Blank Flair-FT ID Enc-0 D-2348
Class	Metric				
Feuilleton	mIoU	74.12 \pm 7.59	44.6 \pm 8.87	40.2 \pm 15.57	49.05 \pm 9.41
	P@60	82.02 \pm 7.24	42.2 \pm 19.95	35.92 \pm 28.4	53.97 \pm 12.42
	P@80	66.45 \pm 14.87	15.72 \pm 20.03	21.02 \pm 22.46	29.95 \pm 23.47
	R@60	97.91 \pm 1.95	97.17 \pm 7.86	77.79 \pm 36.32	100.0 \pm 0.0
	R@80	97.52 \pm 2.26	89.33 \pm 31.46	56.54 \pm 49.2	100.0 \pm 0.0
Weather	mIoU	81.27 \pm 2.18	67.98 \pm 9.26	73.35 \pm 3.89	73.55 \pm 3.55
	P@60	91.08 \pm 4.93	72.79 \pm 10.5	80.41 \pm 8.3	82.29 \pm 5.13
	P@80	66.94 \pm 7.41	48.98 \pm 9.74	49.14 \pm 4.2	58.13 \pm 3.97
	R@60	78.74 \pm 1.98	82.38 \pm 6.14	77.78 \pm 8.04	90.57 \pm 3.64
	R@80	73.06 \pm 2.57	76.09 \pm 6.63	68.53 \pm 9.76	87.18 \pm 4.82
Obituary	mIoU	75.37 \pm 2.98	63.89 \pm 5.48	66.25 \pm 6.75	71.44 \pm 3.26
	P@60	83.37 \pm 4.46	74.67 \pm 7.45	73.43 \pm 10.17	82.19 \pm 5.42
	P@80	67.37 \pm 2.04	37.72 \pm 7.43	47.02 \pm 14.16	62.01 \pm 5.34
	R@60	93.02 \pm 2.88	82.66 \pm 4.67	86.15 \pm 3.66	88.07 \pm 2.91
	R@80	91.49 \pm 3.67	70.07 \pm 10.25	78.93 \pm 8.1	84.72 \pm 4.15
Stocks	mIoU	83.11 \pm 0.88	69.75 \pm 6.22	79.08 \pm 2.46	78.87 \pm 2.61
	P@60	89.21 \pm 1.42	77.6 \pm 5.99	86.59 \pm 2.83	86.85 \pm 2.96
	P@80	80.51 \pm 2.19	58.11 \pm 7.92	73.48 \pm 4.02	74.07 \pm 3.9
	R@60	93.78 \pm 0.73	87.94 \pm 2.75	90.34 \pm 1.81	91.71 \pm 1.23
	R@80	93.15 \pm 0.81	84.51 \pm 3.14	88.81 \pm 2.08	90.42 \pm 1.3
Average	mIoU	79.3 \pm 2.29	62.47 \pm 4.14	65.92 \pm 4.7	69.3 \pm 2.25
	P@60	86.86 \pm 2.6	67.93 \pm 4.98	70.56 \pm 7.56	77.27 \pm 3.03
	P@80	72.29 \pm 3.74	43.02 \pm 6.24	51.09 \pm 6.74	57.93 \pm 5.73
	R@60	90.64 \pm 1.03	86.54 \pm 3.6	86.82 \pm 2.94	91.81 \pm 1.82
	R@80	88.94 \pm 1.4	80.38 \pm 4.14	82.67 \pm 3.46	89.26 \pm 2.78

TABLE 4.6: Results of the different metrics for for the blank images experiment.

The lower results for Fasttext are certainly explained by the 25% miss rate of the words coming from OCR mistakes and completely unknown words. This means that one word out of four is mapped to a blank input, which penalizes the model. The higher ranking of the stacked embeddings suggest that they have a higher representation power, combining the strengths of the two embeddings.

Other metrics

The precision and recall are also considered for this experiment. The metrics are reported in Table 4.6, for each experiment and class, its mIoU is again reported for comparison purposes.

Using these other metrics, the superiority of stacked embeddings is confirmed. It is interesting to notice that this model manages to reach 100% recall for *Feuilleton*, which means that even though it is not precise, it was still able to learn features common to all *Feuilleton* (and some other parts of the newspapers).

The other thing to notice is that the blank model is again very competitive versus Vanilla at precision with a threshold of 60 for *Obituary* and *Stocks*, meaning that these two classes have textual features that makes them distinct enough from the rest. This was already verified for



FIGURE 4.3: Examples of *Weather* that were only detected by blank embeddings, no other models managing to find them.

Stocks where, as shown in picture 3.7, the numbers are distinct with only three dimensions of the embeddings.

The recall score for *Weather* is also very high, showing that the network manages to find more difficult examples of this class, particularly one that are text based and only present during a small time period. Two examples are shown in Figure 4.3.

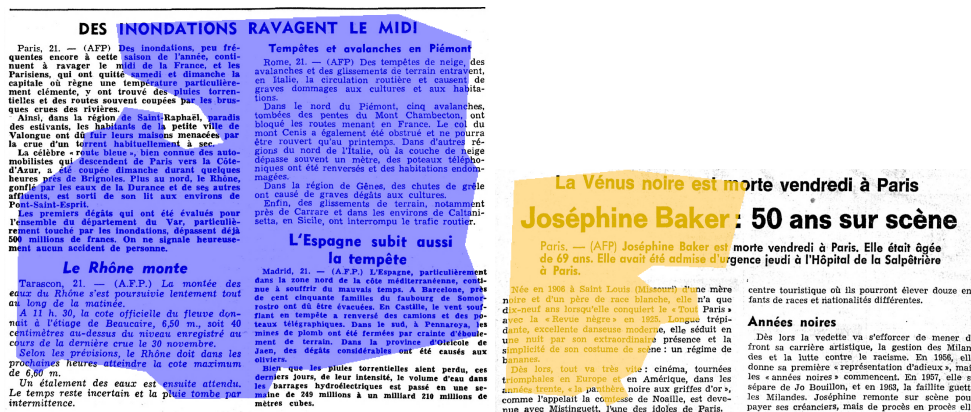


FIGURE 4.4: Examples where the text is related to a class, but not the layout. On the left, an article about floods is detected as a *Weather*. On the right an article about the death of an artist is detected as an *Obituary*

One of the reason of the drop in precision can be found in the presence of false positives that are close to the detected class in term of text, but not in term of layout. Two examples are shown in figure 4.4. In each case the articles have related text, but are not part of the classes. These examples were never detected as false positive when also using the images as input. It suggests that the models with blank images focus more on the textual meaning of the embeddings than on the layout.

Experiment summary

Results are better than expected, particularly considering the quality of the OCR and the fact that the network was not optimized for non-image inputs. It is interesting to see that the hierarchy of embeddings that comes from natural language processing is respected. It means that there is a gain to have both non-contextual and contextual embeddings and let the network pick the best.

The higher recall for *Weather* was particularly useful during the annotation phase, where examples purely textual are harder to find. This network made it possible to correct several mistakes in the annotations.

4.3.3 Embeddings transformation

Experiment description

This experiment shows the effect of dimensionality reduction. Two dimensionality reduction methods are considered:

1. PCA that uses mathematical property of the data to reduce its dimensions, denoted "PCA" in the models
2. An end-to-end reduction that is learned from the data for the task using 1D convolutions, denoted "Conv1D" in the models.

The three embeddings types are again considered, but this time only with an input at the end of the decoder. Inputting the smaller dimension vectors at the beginning of the decoder resulted in bad performances.

Results and discussion

Experiment / Class	Feuilleton	Weather	Obituary	Stocks	Average
Vanilla	74.12 ± 7.59	81.27 ± 2.18	75.37 ± 2.98	83.11 ± 0.88	79.3 ± 2.29
Flair-FT ID Enc-0 D-2348	76.73 ± 5.9	81.38 ± 3.34	83.58 ± 2.02	84.43 ± 1.84	82.16 ± 1.72
Fasttext PCA Dec-1 D-32	77.43 ± 5.15	82.18 ± 2.12	74.86 ± 6.86	82.51 ± 1.83	79.76 ± 1.61
Fasttext Conv1D Dec-1 D-32	76.21 ± 2.86	82.73 ± 1.26	76.52 ± 5.77	82.47 ± 1.39	79.97 ± 1.7
Flair PCA Dec-1 D-256	80.9 ± 2.76	82.9 ± 0.96	75.54 ± 2.88	82.95 ± 2.09	80.87 ± 1.53
Flair Conv1D Dec-1 D-256	78.08 ± 7.1	83.44 ± 2.31	79.45 ± 1.41	83.22 ± 1.98	81.35 ± 1.71
Flair-FT PCA Dec-1 D-256	76.53 ± 6.76	82.2 ± 2.69	75.6 ± 2.39	83.11 ± 2.13	79.94 ± 2.03
Flair-FT Conv1D Dec-1 D-256	78.94 ± 6.47	83.18 ± 1.88	79.61 ± 1.75	82.9 ± 0.88	81.44 ± 1.76

TABLE 4.7: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %). "Flair-FT ID Enc-0 D-2348" is reported again for comparison purposes.

The results are presented in Table 4.7 and in Figure 4.5.

Overall, the results are lower than when not doing any transformation of the embeddings. However, they are still always higher than Vanilla.

For *Feuilleton*, the results are better with a transformation of the embeddings. Particularly for "Flair PCA Dec-1 D-256", where there is a significant increase compared to Vanilla of between

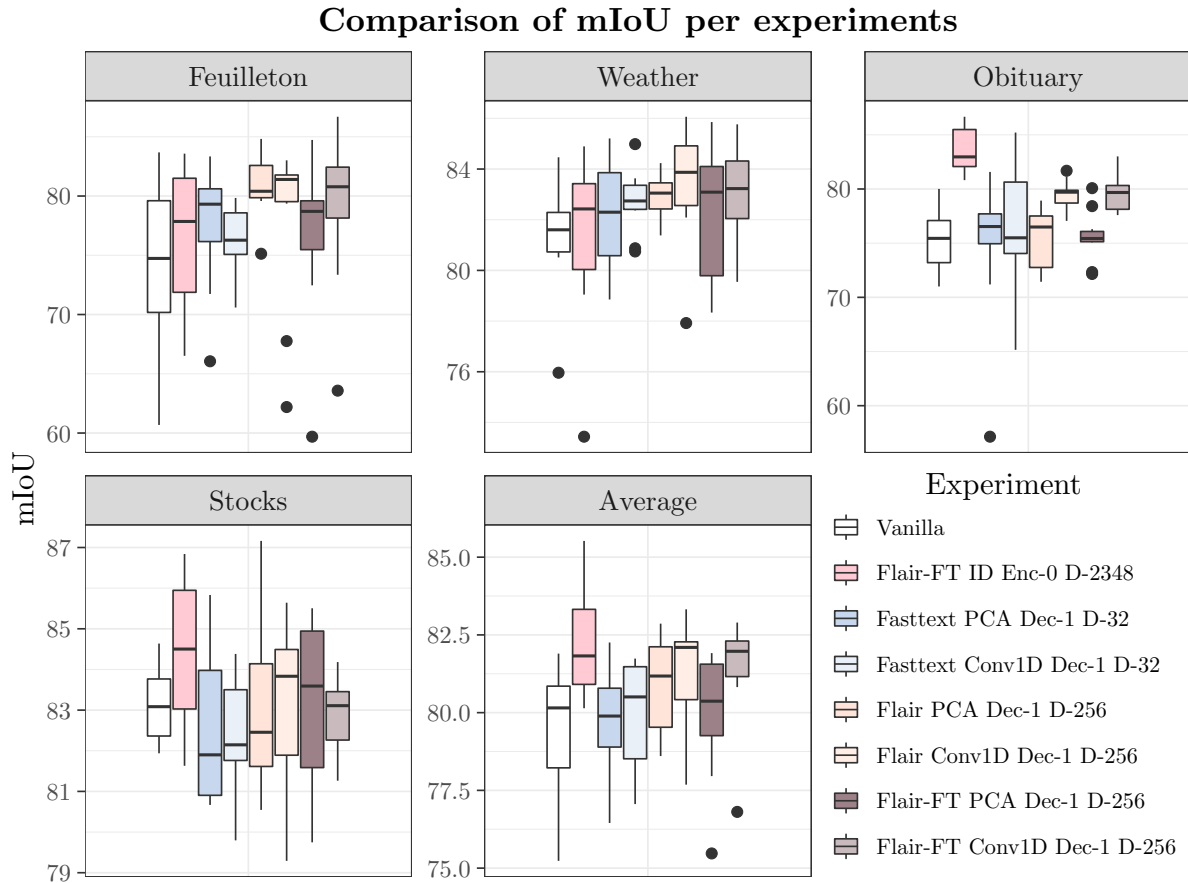


FIGURE 4.5: Box plot of the results with different dimensionality reduction. “Flair-FT ID Enc-0 D-2348” is reported again for comparison purposes.

1.2% and 12.4% at 95% confidence interval. This shows that the 256 most important dimensions are enough to capture most of the textual features of the *Feuilleton*.

For *Weather* the results are better with an end-to-end transformation. This result is significant for “Flair Conv1D Dec-1 D-256” vs Vanilla, where the mean is higher between 0.1% and 4.3% at a 95% confidence interval. The fact that the end-to-end transformation is always a bit higher than PCA suggests that the representation needs to be learned. This would mean that the 256 most important dimensions obtained with PCA are not as good as learned projections to find this class.

For *Obituary*, the results are worse than with no transformation. Even though the two Conv1D models with Flair and Stacked embeddings are significantly higher than Vanilla. The mean of “Flair-FT ID Enc-0 D-2348” is significantly higher than the models using Conv1D, beating them by between 2% and 6% at 95% confidence interval. This is very clear when looking at the box plot. This would mean that obituaries do not benefit to have dimensionality reduction.

For *Stocks*, there is a small decrease compared to the models without transformation. This class is already well segmented by the Vanilla model and the Section 4.3 showed no gain of text embeddings, meaning that the class can be segmented purely visually. This could indicate the reducing the dimensions increases the noise in the features making it harder to learn the necessary visual features.

Experiment		Vanilla	Flair PCA Dec-1 D-256	Flair Conv1D Dec-1 D-256	Flair-FT PCA Dec-1 D-256	Flair-FT Conv1D Dec-1 D-256
Class	Metric					
Feuilleton	mIoU	74.12 \pm 7.59	80.9 \pm 2.76	78.08 \pm 7.1	76.53 \pm 6.76	78.94 \pm 6.47
	P@60	82.02 \pm 7.24	87.07 \pm 3.6	82.89 \pm 7.85	82.83 \pm 6.57	84.34 \pm 6.85
	P@80	66.45 \pm 14.87	76.73 \pm 6.86	80.01 \pm 8.1	75.88 \pm 10.34	81.67 \pm 7.43
	R@60	97.91 \pm 1.95	97.58 \pm 1.27	99.13 \pm 1.39	97.94 \pm 1.42	99.13 \pm 1.41
	R@80	97.52 \pm 2.26	97.27 \pm 1.47	99.1 \pm 1.45	97.74 \pm 1.58	99.11 \pm 1.44
Weather	mIoU	81.27 \pm 2.18	82.9 \pm 0.96	83.44 \pm 2.31	82.2 \pm 2.69	83.18 \pm 1.88
	P@60	91.08 \pm 4.93	93.14 \pm 1.9	94.28 \pm 4.24	93.46 \pm 3.44	93.78 \pm 3.77
	P@80	66.94 \pm 7.41	74.18 \pm 4.16	74.3 \pm 5.82	71.83 \pm 5.31	71.68 \pm 6.4
	R@60	78.74 \pm 1.98	80.62 \pm 1.7	81.29 \pm 3.59	80.55 \pm 1.63	80.58 \pm 2.37
	R@80	73.06 \pm 2.57	76.78 \pm 2.25	77.41 \pm 4.17	76.05 \pm 2.17	75.97 \pm 3.12
Obituary	mIoU	75.37 \pm 2.98	75.54 \pm 2.88	79.45 \pm 1.41	75.6 \pm 2.39	79.61 \pm 1.75
	P@60	83.37 \pm 4.46	83.8 \pm 3.19	89.65 \pm 2.95	83.43 \pm 2.85	89.58 \pm 3.71
	P@80	67.37 \pm 2.04	68.66 \pm 4.29	68.33 \pm 5.88	69.6 \pm 4.55	67.09 \pm 5.33
	R@60	93.02 \pm 2.88	94.72 \pm 2.62	94.76 \pm 3.96	94.07 \pm 2.86	95.82 \pm 3.4
	R@80	91.49 \pm 3.67	93.69 \pm 3.1	93.29 \pm 5.19	92.91 \pm 3.48	94.61 \pm 4.1
Stocks	mIoU	83.11 \pm 0.88	82.95 \pm 2.09	83.22 \pm 1.98	83.11 \pm 2.13	82.9 \pm 0.88
	P@60	89.21 \pm 1.42	89.44 \pm 1.48	89.85 \pm 1.95	89.88 \pm 1.51	88.34 \pm 0.93
	P@80	80.51 \pm 2.19	80.94 \pm 1.82	80.94 \pm 2.28	81.96 \pm 3.2	81.07 \pm 1.25
	R@60	93.78 \pm 0.73	94.12 \pm 0.53	93.88 \pm 0.06	93.8 \pm 0.33	94.09 \pm 0.53
	R@80	93.15 \pm 0.81	93.55 \pm 0.52	93.25 \pm 0.16	93.23 \pm 0.44	93.6 \pm 0.61
Average	mIoU	79.3 \pm 2.29	80.87 \pm 1.53	81.35 \pm 1.71	79.94 \pm 2.03	81.44 \pm 1.76
	P@60	86.86 \pm 2.6	88.47 \pm 1.43	89.23 \pm 1.78	87.76 \pm 2.09	88.85 \pm 1.84
	P@80	72.29 \pm 3.74	76.05 \pm 1.57	76.56 \pm 1.91	75.94 \pm 2.88	76.23 \pm 1.81
	R@60	90.64 \pm 1.03	91.51 \pm 0.73	91.87 \pm 1.34	91.31 \pm 0.55	92.0 \pm 1.09
	R@80	88.94 \pm 1.4	90.27 \pm 0.79	90.64 \pm 1.58	90.08 \pm 0.64	90.8 \pm 1.19

TABLE 4.8: Results of the different metrics for the embeddings transformation experiment.

Other metrics

This section explores how the other metrics behave with the different models. It was chosen to not report Fasttext results as they were below the other embeddings types. The metrics are reported in Table 4.8, for each experiment and class, its mIoU is again reported for comparison purposes.

The results with the other metrics confirm the superiority of Flair and Stacked embeddings with Conv1D dimensionality reduction. Overall the results are very similar than in the embeddings and input location experiment with no significant improvement and the same drop in precision when considering the two different thresholds.

The only noticable gain is the precision for *Weather* at threshold 60 which is higher than without transformation reaching more than 94% for “Flair Conv1D Dec-1 D-256”. This makes the result very good for downstream tasks that require little noise.

Experiment summary

There is no strong gain in using dimensionality reduction. This could be because the network is already able to process the raw embeddings and reduce their dimensions. Meaning that a pre-processing step is not necessary.

Overall, end-to-end dimensionality reduction seems to be a bit better than using PCA. This result could be improved with more data, since the dimensionality reduction has to be learned.

4.3.4 Discussion

In these experiments, it was shown that the text embeddings maps have strong representation power by themselves. It was also shown that when combining them with the visual features, it created models that performed better than the baseline. The gain depending on the class. For *Stocks*, there is little gain, indicating that this class has distinct enough visual features. The biggest improvements were with *Obituary*, because the mixed models were more robust to false positives. The gain was also important with *Weather*, because the mixed models managed to detect more examples, particularly ones that are entirely textual.

Pre-processing the embeddings by reducing the dimension showed no real improvement over using them directly. This could be either because of a lack of data to learn a meaningful representation or because the number of dimensions is not adequate. This would also indicate the the models are already able to process the embeddings maps and learn a representation suited for the task without the need for pre-processing.

For practical purposes, the scores are promising. Indeed, the maximum precision at threshold 60 on most of the classes was of around 90%, meaning that only one result out of ten is a false positive. The maximum precision score at threshold 80 oscillates between 70% and 80%, making it harder to use for any task that require this kind of threshold. However the recall scores are higher, between 94% and 100% for the best models. These models would thus be ready for any task requiring a high recall.

4.4 Generalization experiments

In this section, the experiments assesses the generalization ability of the models. In particular it shows the performances change when training with less data, when training with no data from a particular time period and when testing models trained on one particular newspaper on other newspapers.

Three models from the previous experiment have been selected for the comparison:

- Vanilla in order to have a baseline.
- Stacked embeddings with no dimensionality reduction, input at the beginning of the encoder, because it was the overall best performing model.
- Stacked embeddings with an end-to-end dimensionality reduction, because it was the best dimensionality reduction technique.

4.4.1 Amount of training data

Experiment description

This experiment shows how the amount of training data influences the performances. In the first experiments, the split between training and testing sets was of 70% and 30%, respectively, now it is 40% training and 60% testing. This means that the number of training examples decrease from 1387 to 792 and the testing ones increase from 416 to 1190. The actual distribution is shown in Table 4.9. As previously, the ratio of each class is similar between training and testing.

Class	Train count	Test count	Train ratio (%)	Test ratio (%)
Feuilleton	56	81	7.07	6.81
Weather forecast	61	95	7.70	7.98
Obituary	59	94	7.45	7.90
Stock exchange table	92	183	11.62	15.38
Empty	578	815	72.98	68.49

TABLE 4.9: Distribution of the classes for the training and testing sets.

Results and discussion

Experiment/Class	Feuilleton	Weather	Obituary	Stocks	Average
All training data					
Vanilla	74.12 ± 7.59	81.27 ± 2.18	75.37 ± 2.98	83.11 ± 0.88	79.3 ± 2.29
Flair-FT ID Enc-0 D-2348	76.73 ± 5.9	81.38 ± 3.34	83.58 ± 2.02	84.43 ± 1.84	82.16 ± 1.72
Flair-FT Conv1D Dec-1 D-256	78.94 ± 6.47	83.18 ± 1.88	79.61 ± 1.75	82.9 ± 0.88	81.44 ± 1.76
Less training data					
Vanilla	70.27 ± 2.79	69.67 ± 2.76	65.88 ± 6.48	74.73 ± 2.25	70.8 ± 2.32
Flair-FT ID Enc-0 D-2348	49.36 ± 14.22	66.71 ± 3.57	71.2 ± 4.03	77.3 ± 1.07	68.22 ± 2.84
Flair-FT Conv1D Dec-1 D-256	70.74 ± 6.71	72.51 ± 2.32	70.7 ± 1.66	75.54 ± 1.4	72.83 ± 1.9

TABLE 4.10: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %). The results from the previous experiments are again reported for comparison purposes.

The results are presented in Table 4.10 and in Figure 4.6.

When comparing the models trained with more and less data there is an overall drop in performances. The best model on the averaged mean loses around 10%. This shows the importance of having a good amount of training samples. This is partly explained by the fact that the layout and content of the classes change a lot through time. It thus means that with less data, the model will have even less training sample of each type.

The performances of the three models using less training data show an overall hierarchy:

1. Less data Flair-FT Conv1D Dec-1 D-256
2. Less data Vanilla
3. Less data Flair-FT ID Enc-0 D-2348

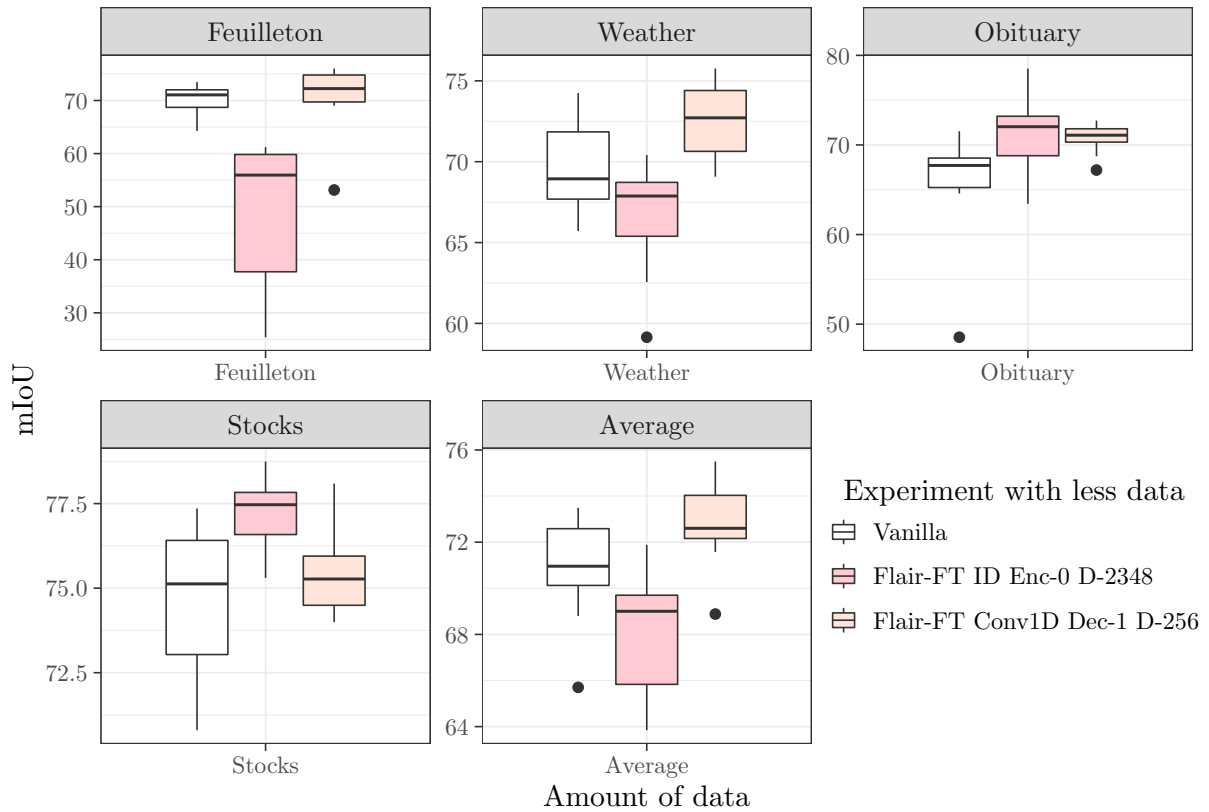


FIGURE 4.6: Box plot of the results with less training data.

This shows that the non processed embeddings model needs more data to learn useful representation, certainly because it starts from a larger dimension. In particular, it performs quite good on *Obituary* and *Stocks* and poorly on *Feuilleton* and *Weather*, showing that it is hard to perform well on all classes. This is quite logical since the embeddings map in this case has to pass through the whole network with a lot of parameters not dedicated to the transformation of the map. It is thus harder to learn a good representation that is useful to all the classes with little data.

Contrary to the non processed embeddings, the end-to-end approach has a dedicated set of weights to create the transformation. This transformation seems easier to learn with less data and is an improvement over the Vanilla approach for all classes. In particular for *Weather* and *Obituary* where the improvements are significant. For *Weather* the improvement is between 0.6% and 2.9% at 95% confidence interval and for *Obituary* between 0.9% and 4.3%.

Other metrics

The precision and recall are also considered for this experiment. The metrics are reported in Table 4.11, for each experiment and class, its mIoU is again reported for comparison purposes.

When compared with results of Table 4.4 and Table 4.8 the precision and recall are both lower. This shows the importance of the number of training samples.

Experiment		Less data Vanilla	Less data Flair-FT ID Enc-0 D-2348	Less data Flair-FT Conv1D Dec-1 D-256
Class	Metric			
Feuilleton	mIoU	70.27 \pm 2.79	49.36 \pm 14.22	70.74 \pm 6.71
	P@60	76.85 \pm 4.52	44.35 \pm 28.58	74.87 \pm 8.06
	P@80	56.42 \pm 7.5	26.14 \pm 20.36	65.13 \pm 6.65
	R@60	93.85 \pm 2.42	81.91 \pm 34.16	95.91 \pm 1.97
	R@80	91.45 \pm 4.4	68.23 \pm 47.14	95.32 \pm 2.3
Weather	mIoU	69.67 \pm 2.76	66.71 \pm 3.57	72.51 \pm 2.32
	P@60	73.46 \pm 5.62	73.36 \pm 5.02	78.34 \pm 4.07
	P@80	40.47 \pm 7.0	41.28 \pm 4.71	49.38 \pm 5.98
	R@60	61.5 \pm 2.01	67.07 \pm 4.32	65.11 \pm 2.92
	R@80	46.59 \pm 3.76	53.43 \pm 5.07	53.96 \pm 3.45
Obituary	mIoU	65.88 \pm 6.48	71.2 \pm 4.03	70.7 \pm 1.66
	P@60	72.27 \pm 7.46	75.92 \pm 4.95	76.53 \pm 2.78
	P@80	52.62 \pm 7.11	61.94 \pm 5.14	59.92 \pm 2.21
	R@60	92.04 \pm 2.54	93.24 \pm 2.36	93.89 \pm 1.98
	R@80	89.3 \pm 3.53	91.82 \pm 2.87	92.32 \pm 2.54
Stocks	mIoU	74.73 \pm 2.25	77.3 \pm 1.07	75.54 \pm 1.4
	P@60	83.49 \pm 2.27	88.27 \pm 1.41	84.35 \pm 1.19
	P@80	63.44 \pm 3.75	64.29 \pm 3.83	64.96 \pm 2.93
	R@60	90.21 \pm 1.61	89.99 \pm 1.51	90.09 \pm 1.44
	R@80	87.47 \pm 2.18	86.73 \pm 2.11	87.5 \pm 1.82
Average	mIoU	70.8 \pm 2.32	68.22 \pm 2.84	72.83 \pm 1.9
	P@60	77.81 \pm 3.16	73.78 \pm 5.75	79.5 \pm 2.09
	P@80	55.95 \pm 3.81	51.92 \pm 4.2	61.32 \pm 1.65
	R@60	85.52 \pm 0.67	86.59 \pm 1.82	86.8 \pm 1.09
	R@80	80.91 \pm 1.18	81.96 \pm 2.37	83.53 \pm 1.22

TABLE 4.11: Results of the different metrics for the less training data experiment.

For example in the previous experiments, the precision at threshold 60 for the *Weather* was of around 94% for Flair-FT Conv1D Dec-1 D-256, which is a precision that is good for most downstream tasks, however with less training data, it drops to 78% which is pretty bad.

The drop in term of recall for *Weather* is also important with more than 20% decrease between the best of each approach. This means that a lot of examples are missed, certainly because there were not enough similar samples in the training data.

Experiment summary

This experiments shows the importance of the number of training samples. When trained with less data, the models have more trouble generalizing, resulting in more false positive and false negatives.

The usage of embeddings can help with this lack of data, particularly using end-to-end representation learning. This shows that the embeddings give enough additional signal that the model can pick up in order to better generalize and retrieve more examples.

4.4.2 Generalization across time

Experiment description

Class	Train count	Test count	Train ratio (%)	Test ratio (%)
Feuilleton	134	3	9.61	0.51
Weather forecast	132	24	9.47	4.08
Obituary	124	29	8.90	4.93
Stock exchange table	211	64	15.14	10.88
Empty	923	470	66.21	79.93

TABLE 4.12: Distribution of the classes for the training and testing sets.

This experiment measures how well the models generalize when trained only on partial data across time. The time period that is removed is chosen using the study presented in [55]. In this study, the authors analyzed the layout of *Journal de Genève* and clustered them by time period. The period 1969-1991 was chosen because it was a period of great shift in term of layout and it allowed to have a similar number of examples in the training and testing set. The goal of this separation is that the testing set contains visually distinct examples of the classes in order to assess the generalization of the network.

The training data comprises 1394 examples from the period 1826-1968 and the period 1992-1998. It thus has both examples from before and after the removed period. The testing data has 588 examples of the period 1969-1991. The distribution by class is shown in Table 4.12. This time the ratio of each class is not well balanced between training and testing. In particular, the training set contains more examples of each class.

Results and discussion

The results are presented in Table 4.13 and in Figure 4.7.

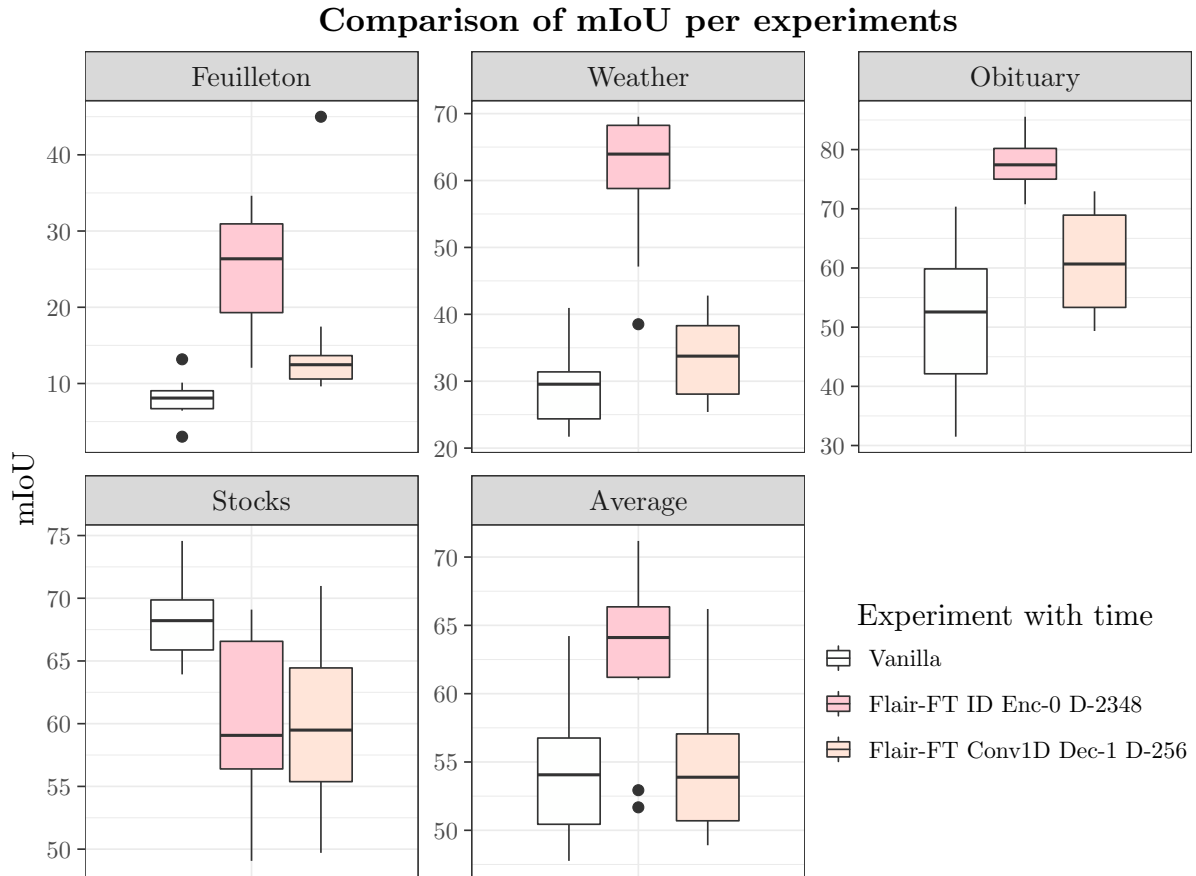


FIGURE 4.7: Box plot of the results for the models testing time generalization.

Experiment / Class	Feuilleton	Weather	Obituary	Stocks	Average
Time Vanilla	8.0 ± 2.66	29.44 ± 6.28	51.29 ± 12.88	68.3 ± 3.31	54.65 ± 5.47
Time Flair-FT ID Enc-0 D-2348	25.08 ± 7.37	60.65 ± 10.27	77.52 ± 4.41	60.17 ± 7.42	62.84 ± 6.37
Time Flair-FT Conv1D Dec-1 D-256	15.54 ± 10.6	33.58 ± 6.02	61.11 ± 8.96	59.8 ± 6.29	54.95 ± 5.35

TABLE 4.13: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %).

Even though the number of training sample for each class was larger in this experience than in the experiments of Section 4.3. The mIoU score is much lower for every classes. This indicates that the testing set of this experiment is quite different from the training set.

By looking at the box-plots of Figure 4.7, it is clear that the model with Stacked embeddings and no dimensionality reduction performs much better than the other two, except for *Stocks*. This model's difference of mean with the other two is significant for *Feuilleton* (gain of between 11% and 22% versus Vanilla at 95% confidence), for *Weather* (gain of between 23% and 39% versus Vanilla at 95% confidence), for *Obituary* (gain of between 16% and 35% versus Vanilla at 95% confidence) and on average (gain of between 3% and 14% versus Vanilla at 95% confidence).

The drop of performances for *Stocks* for the two models that are using textual features is significant. The Vanilla model has a mean higher between 2.5% and 13.7% versus "Flair-FT ID Enc-0 D-2348" and between 3.7% and 13.3% versus "Flair-FT Conv1D Dec-1 256". This could mean that *Stocks* have a more stable visual form than the other classes and that its textual particularities are not enough to generalize.

Experiment		Time Vanilla	Time Flair-FT ID Enc-0 D-2348	Time Flair-FT Conv1D Dec-1 D-256
Class	Metric			
Feuilleton	mIoU	8.0 ± 2.66	25.08 ± 7.37	15.54 ± 10.6
	P@60	8.4 ± 2.52	20.03 ± 7.23	16.04 ± 13.18
	P@80	8.0 ± 3.44	20.03 ± 7.23	7.01 ± 6.14
	R@60	53.33 ± 25.82	95.0 ± 15.81	55.0 ± 29.45
	R@80	50.0 ± 30.43	95.0 ± 15.81	33.33 ± 33.33
Weather	mIoU	29.44 ± 6.28	60.65 ± 10.27	33.58 ± 6.02
	P@60	21.54 ± 8.03	60.4 ± 19.01	23.06 ± 3.96
	P@80	12.28 ± 7.77	36.61 ± 12.75	16.99 ± 7.63
	R@60	13.8 ± 3.56	68.63 ± 15.93	15.61 ± 1.95
	R@80	8.73 ± 5.48	57.76 ± 18.56	11.84 ± 4.82
Obituary	mIoU	51.29 ± 12.88	77.52 ± 4.41	61.11 ± 8.96
	P@60	48.6 ± 23.89	86.47 ± 4.82	56.86 ± 21.53
	P@80	23.92 ± 16.58	78.09 ± 6.05	25.7 ± 9.4
	R@60	95.42 ± 10.84	100.0 ± 0.0	98.18 ± 3.83
	R@80	90.0 ± 24.15	100.0 ± 0.0	95.83 ± 9.0
Stocks	mIoU	68.3 ± 3.31	60.17 ± 7.42	59.8 ± 6.29
	P@60	72.62 ± 4.21	68.47 ± 5.47	66.5 ± 6.71
	P@80	61.91 ± 4.14	42.62 ± 18.78	36.89 ± 19.23
	R@60	93.82 ± 2.39	91.56 ± 2.08	94.25 ± 3.3
	R@80	92.87 ± 2.58	84.98 ± 7.87	87.85 ± 11.1
Average	mIoU	54.65 ± 5.47	62.84 ± 6.37	54.95 ± 5.35
	P@60	55.9 ± 7.9	69.48 ± 6.32	57.05 ± 6.52
	P@80	42.67 ± 5.49	49.23 ± 13.07	30.78 ± 12.8
	R@60	78.11 ± 4.53	90.16 ± 2.74	79.01 ± 2.06
	R@80	73.27 ± 4.88	85.83 ± 5.94	64.96 ± 9.19

TABLE 4.14: Results of the different metrics for the time experiment.

The performances of the stacked embeddings with an end-to-end dimensionality reduction are lower than when using non-processed embeddings map. This indicates that the Conv1D has more trouble generalizing and that it may suffer from some form of over-fitting in the way it learns its dimensionality reduction. The previous experiment showed that even with little data it was able to learn meaningful representations, but it seems to be not the case when confronted to unseen examples.

Other metrics

The precision and recall are also reported for this experiment in Table 4.11. For each experiment and class, its mIoU is again reported for comparison purposes. The results for the other metrics are similar to the one discussed in the previous section.

It is interesting to notice that for all classes, except for *Weather*, the recall is quite high for the "Flair-FT ID Enc-0 D-2348" model. Particularly for *Obituary* where the model achieve a perfect recall score. This means that the main reason for a drop in mIoU is mainly false positives, thus the model is able to learn features distinct to all the classes. One particular use case that could benefit from this model would be when creating new annotations. Indeed, it is possible to start the annotation process by only considering a time period, train a model on it and make new

predictions on other time periods. Because of the high recall, the annotation tasks would be focused on filtering the false positive. This would make the annotation process faster.

The bad recall results for *Weather* with every models hints that this class changes a lot during the particular time period. It is thus much harder to achieve a good recall.

Experiment summary

The performance drop between the models of Section 4.3 and this model shows that there is a gain to have an uniform annotation across time. This also supports the theory that newspapers change a lot throughout time, hence the performance drop when removing a time period from the training set. It also backs the fact that it is difficult to know at which time interval to annotate in order to be able to have a model general enough to work on all time periods.

The performance gain when using textual features is however very promising. It supports the theory that textual features are more stable than visual ones.

Even though the general performances are not satisfying for most practical purpose, the models could still be useful in an annotation scenario.

4.4.3 Generalization across newspaper

Experiment description

Class	GDL	IMP
Feuilleton	108	103
Weather forecast	68	41
Obituary	69	102
Stock exchange table	135	79
Empty	697	1326

TABLE 4.15: Distribution of the classes for the *Gazette de Lausanne* and *Impartial*.

This experiments tests similar properties of the models as the previous one. However, this time rather than considering a difference of time, a difference of newspaper is examined.

The experiments uses the models trained in Section 4.3 using images from *Journal de Genève*. The models are taken as is and are used to predict the classes of images of *Gazette de Lausanne* and *Impartial*. For reminder, there is 1008 annotations of *Gazette de Lausanne* and 1634 for *Impartial*. The classes repartition is shown in Table 4.15

As discussed previously, the results are expected to be different between *Gazette de Lausanne* and *Impartial*. Indeed, *Journal de Genève* is closer to *Gazette de Lausanne* than to *Impartial* both in term of time span and in geographical region. The tasks is difficult, since it is very likely that unseen examples will show up.

Results and discussion

The results for this experiment are reported in Table 4.16 and in Figure 4.8.

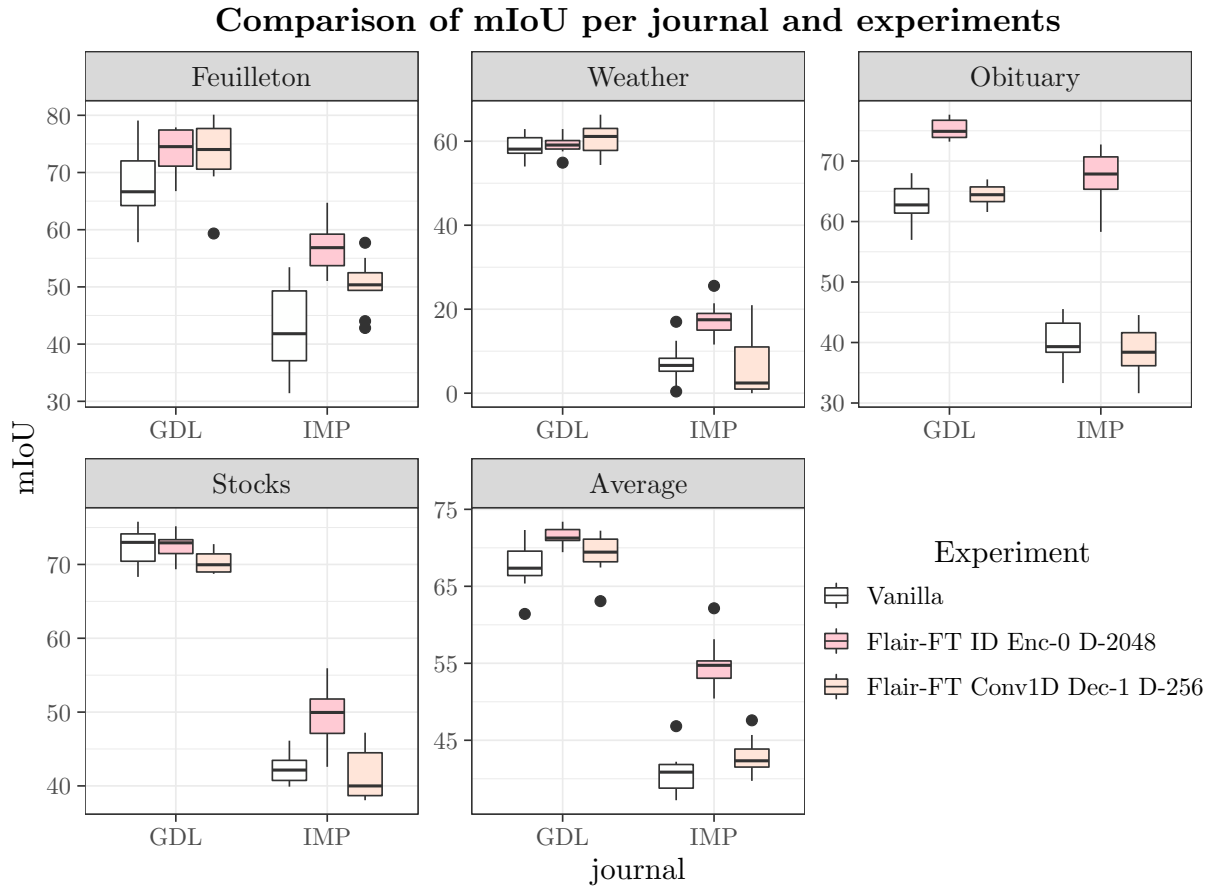


FIGURE 4.8: Box plot of the results of the generalization on two other newspapers.

There is a large difference in term of results between *Gazette de Lausanne* and *Impartial*, the former having higher score than the latter. This confirms that *Journal de Genève* is closer to *Gazette de Lausanne* than to *Impartial*. In particular the difference of performances of more than 40% for *Weather* indicates that *Journal de Genève* and *Gazette de Lausanne* have relatively similar weather forecasts, whereas the ones of *Impartial* are very different. The only similar results are for *Obituaries* predicted by "Flair-FT ID Enc-0 D-2048", showing that this class more stable in term of textual features. The fact that the overall difference is very large would show that the textual features used by this model are better at generalization than either visual features or learned embeddings projection.

Now considering only the results for *Gazette de Lausanne*, there is a lower difference of performances between textual and non textual models than there was with the time experiment. This could be due to the fact that the two newspapers are quite close across all time periods. Also, because the difference with Vanilla is not very large, it would mean that they share a common layout.

The results of the *Impartial* are very low in general. However, the fact that "Flair-FT ID Enc-0 D-2048" is consistently higher than the other two confirms the fact that it is better at generalization when there is little in common in term of layout, as this was the case in the time experiment.

Experiment / Class	Feuilleton	Weather	Obituary	Stocks	Average
GDL Vanilla	67.79 ± 6.62	58.6 ± 3.0	63.06 ± 3.26	72.38 ± 2.35	67.59 ± 3.12
GDL Flair-FT ID Enc-0 D-2048	73.81 ± 4.08	59.16 ± 2.22	75.32 ± 1.69	72.65 ± 1.77	71.54 ± 1.21
GDL Flair-FT Conv1D Dec-1 D-256	73.43 ± 6.23	60.5 ± 3.59	64.39 ± 1.7	70.26 ± 1.42	69.13 ± 2.67
IMP Vanilla	42.45 ± 7.86	7.04 ± 4.92	40.14 ± 3.81	42.45 ± 2.08	40.71 ± 2.82
IMP Flair-FT ID Enc-0 D-2048	56.7 ± 4.23	17.53 ± 4.11	67.36 ± 4.43	49.46 ± 3.84	54.97 ± 3.25
IMP Flair-FT Conv1D Dec-1 D-256	50.4 ± 4.52	6.02 ± 7.43	38.46 ± 3.97	41.61 ± 3.5	42.94 ± 2.35

TABLE 4.16: Results for the mIoU metric, results are reported under the form mean \pm standard deviation (in %).

Other metrics

The precision and recall for this experiments are reported in Table 4.17 for *Gazette de Lausanne* and in Table 4.18 for *Impartial*.

For *Gazette de Lausanne* the results are similar to the ones of the time experiment. The annotation use case could also be exploited. However, for *Impartial*, the results are even lower and thus become harder to exploit for the annotation use case. Particularly for *Weather* where the recall is less than 7%.

Experiment summary

This experiment shows that it is not enough to train on one newspaper and hope to be able to generalize well on other newspapers, except if the newspaper is very close in term of layout.

It is however interesting that the textual features improve the generalization, particularly for *Obituary*. This again supports the hypotheses that textual features are more stable than visual features.

4.4.4 Discussion

Overall all the models have trouble generalizing well. This shows the complexity of the newspaper object. Its variety of layout and text used makes it difficult to generalize across time and newspaper and it requires a large amount of annotated data to train models.

The fact that the text embeddings consistently improved the generalization really supports the hypothesis that the text is more stable than the visual and that it makes sense to use it under the form of a textual embeddings map.

The difficulty to generalize across time and newspaper show that this kind of model is not ready to be deployed at large scale for solving the semantic segmentation of newspapers. However, their performances in term of precision and recall makes them good candidate to generate annotations in a more efficient way, by having them pre-annotate the images and then verify them manually.

Experiment		GDL Vanilla	GDL Flair-FT ID Enc-0 D-2348	GDL Flair-FT Conv1D Dec-1 D-256
Class	Metric			
Feuilleton	mIoU	67.79 \pm 6.62	73.81 \pm 4.08	73.43 \pm 6.23
	P@60	69.8 \pm 7.7	77.77 \pm 6.02	77.24 \pm 6.6
	P@80	55.55 \pm 9.55	61.75 \pm 11.66	66.43 \pm 6.19
	R@60	95.64 \pm 1.84	96.49 \pm 1.03	97.42 \pm 0.81
	R@80	94.38 \pm 2.98	95.45 \pm 1.54	97.01 \pm 0.98
Weather	mIoU	58.6 \pm 3.0	59.16 \pm 2.22	60.5 \pm 3.59
	P@60	55.24 \pm 5.29	56.29 \pm 4.38	58.49 \pm 5.13
	P@80	26.9 \pm 3.3	34.9 \pm 5.78	29.76 \pm 3.94
	R@60	50.61 \pm 2.68	59.0 \pm 5.5	50.98 \pm 3.05
	R@80	33.25 \pm 2.26	47.07 \pm 7.19	34.55 \pm 2.38
Obituary	mIoU	63.06 \pm 3.26	75.32 \pm 1.69	64.39 \pm 1.7
	P@60	64.61 \pm 4.8	82.7 \pm 1.68	65.32 \pm 2.45
	P@80	52.01 \pm 7.21	63.62 \pm 5.43	43.3 \pm 5.26
	R@60	65.46 \pm 5.54	89.92 \pm 3.44	66.92 \pm 3.39
	R@80	60.26 \pm 7.12	87.11 \pm 4.85	57.15 \pm 4.95
Stocks	mIoU	72.38 \pm 2.35	72.65 \pm 1.77	70.26 \pm 1.42
	P@60	76.83 \pm 3.0	78.02 \pm 1.79	73.49 \pm 1.83
	P@80	60.24 \pm 3.47	57.93 \pm 4.97	57.26 \pm 1.93
	R@60	94.74 \pm 1.32	92.51 \pm 2.96	95.47 \pm 1.82
	R@80	93.38 \pm 1.66	90.15 \pm 3.81	94.26 \pm 2.31
Average	mIoU	67.59 \pm 3.12	71.54 \pm 1.21	69.13 \pm 2.67
	P@60	69.78 \pm 3.73	75.67 \pm 1.61	71.57 \pm 2.98
	P@80	53.12 \pm 3.52	56.89 \pm 3.52	54.78 \pm 2.66
	R@60	82.32 \pm 1.44	87.8 \pm 1.61	83.85 \pm 1.45
	R@80	77.99 \pm 1.67	84.37 \pm 2.19	79.9 \pm 1.64

TABLE 4.17: Results of the different metrics on *Gazette de Lausanne*.

Experiment		IMP Vanilla	IMP Flair-FT ID Enc-0 D-2348	IMP Flair-FT Conv1D Dec-1 D-256
Class	Metric			
Feuilleton	mIoU	42.45 \pm 7.86	56.7 \pm 4.23	50.4 \pm 4.52
	P@60	36.43 \pm 13.01	57.04 \pm 6.5	49.43 \pm 4.56
	P@80	16.2 \pm 10.19	37.98 \pm 11.45	31.77 \pm 7.55
	R@60	52.73 \pm 17.06	82.89 \pm 3.37	76.11 \pm 4.73
	R@80	33.45 \pm 18.94	74.88 \pm 9.4	66.61 \pm 8.14
Weather	mIoU	7.04 \pm 4.92	17.53 \pm 4.11	6.02 \pm 7.43
	P@60	5.33 \pm 8.27	7.66 \pm 5.1	5.0 \pm 11.25
	P@80	0.0 \pm 0.0	1.76 \pm 2.33	0.0 \pm 0.0
	R@60	1.02 \pm 1.32	6.5 \pm 4.62	0.49 \pm 1.03
	R@80	0.0 \pm 0.0	1.59 \pm 2.07	0.0 \pm 0.0
Obituary	mIoU	40.14 \pm 3.81	67.36 \pm 4.43	38.46 \pm 3.97
	P@60	32.55 \pm 6.15	72.57 \pm 4.72	23.46 \pm 8.23
	P@80	15.26 \pm 5.91	54.93 \pm 6.29	10.18 \pm 5.71
	R@60	46.86 \pm 9.5	79.26 \pm 2.01	33.72 \pm 10.44
	R@80	29.53 \pm 11.94	74.15 \pm 3.89	18.31 \pm 10.24
Stocks	mIoU	42.45 \pm 2.08	49.46 \pm 3.84	41.61 \pm 3.5
	P@60	47.23 \pm 2.73	54.74 \pm 4.36	47.23 \pm 4.14
	P@80	32.33 \pm 2.91	40.17 \pm 3.35	32.41 \pm 2.16
	R@60	76.1 \pm 3.7	76.01 \pm 6.53	79.81 \pm 3.79
	R@80	68.55 \pm 4.68	70.17 \pm 6.31	73.14 \pm 4.72
Average	mIoU	40.71 \pm 2.82	54.97 \pm 3.25	42.94 \pm 2.35
	P@60	38.65 \pm 4.45	57.62 \pm 4.17	41.09 \pm 2.16
	P@80	21.86 \pm 3.68	41.43 \pm 4.91	25.99 \pm 1.93
	R@60	49.54 \pm 5.96	71.96 \pm 1.9	54.77 \pm 3.06
	R@80	35.76 \pm 6.26	64.75 \pm 2.97	43.39 \pm 3.43

TABLE 4.18: Results of the different metrics on *Impartial*.

5 Discussion and conclusion

This section sums up the findings of this project and critically discusses the contribution to the field of semantic segmentation and document analysis. Section 5.1 discusses the main findings in link with the two hypotheses formulated in Section 3.1, using the results of the experiments of Section 4. Section 5.2 summarizes the contributions to the semantic segmentation of newspapers while Section 5.3 states some limits. Finally, Section 5.4 explores potential works that could be made to improve the results and make the approach more practically usable.

5.1 Discussion

The first hypothesis tested in Section 4.3 was that *visual and textual features can efficiently segment images of newspapers*. The experiments explored the best combination of type of embeddings, input level in the network and dimensionality reduction.

Considering a mIoU threshold of 60%, the best model reached a precision between 85% and 92%, depending on the class. The best model using textual features improving significantly the baseline for *Obituary* by around 7%. This shows that the performances of the models in term of precision at a low threshold are good for most use cases and that the multimodal approaches are superior to the unimodal one. However, the precision with a high threshold was much lower, losing between 10% and 20%. This means that the models do not manage to generate robust predictions and that post-processing is necessary to use the results.

The recall scores at both thresholds oscillated between 84% and 100% for the best model. This shows that the models are very good at finding features that are common to all members of a particular class, even though it also means that it generates false positives.

The results are thus overall satisfying, validating the hypothesis, even though some results are not robust enough without post-processing in some cases. However, it is difficult to assess their quality without performing downstream tasks or comparing to another study.

The second hypothesis tested in section 4.4 was that *Visual features are less stable than textual features and will be worse at generalization*. The experiments explored how the models behaved when trained with fewer data. They also attested the generalization of the models when tested on a new time period or on new newspapers.

With less training data, the textual model using directly the textual embeddings performed worse than the baseline. The textual model pre-processing the embeddings (by reducing the dimensions) managed to improve the precision at threshold 60 over the baseline, but only by between 0% and 5%. Textual features are thus helping the learning when less data is available. Overall, the results were much lower than with more training data, resulting in a drop of between 4% and 18%. This shows the importance of as much training data as possible.

Testing the model on a new time period or on different newspapers showed that textual features were much better at generalizing with unseen items. The best model increased the precision at threshold 60 by between 4% and 40%, showing a gain of the use of embeddings map.

Overall, the scores were very low and not on par with the ones of the experiments with uniform data.

In summary, it was shown that textual features are more stable and generalize better than a purely visual model. This validates the hypothesis. However, it is important to note that this generalization is still not strong enough to provide scores that make the results usable in downstream tasks.

5.2 Contributions

Text embeddings maps improve the raw performances of newspapers image semantic segmentation.

This finding indicates that there is a gain to perform the semantic classification of document processing not only using the text, but also the images. The reconciliation of the two showed an improvement in both precision and recall. It also showed an improvement in the robustness of the predictions, reaching a better mIoU and making less false positive.

Text embeddings maps improve the generalization.

This indicates that textual features are more stable than visual ones and that they are able to capture representations that generalize better. This ability to generalize is promising, but the decrease in overall scores shows that it still needs improvements.

5.3 Limits

The biggest flaw of this project is that it misses a comparison with other approaches. The results are thus the first ones and are harder to interpret in a larger context. It would have been very beneficial to measure the gain of this approach versus the more classical one that first segments the image, extracts the text and finally classify it.

A limit of the approach presented in this project is that it relies on the pre-existence of an OCR. The question of the availability and the quality of the OCR is thus very central to this method. It would thus be particularly important to assess how the performances of the method are affected by the quality of the OCR used. Moreover, using pre-computed OCR violates the end-to-end principle. Even worse, the OCR is also the product of a segmentation, meaning that the segmentation of this project is done on top of another one.

Finally, another limitation is that the results that are reported are without any post-processing. Depending on the quality of the raw predictions, a post-processing step could drastically improve the different scores. Moreover, the performances that are reported and analyzed do not necessarily reflect the true performances of the models with proper post-processing.

5.4 Improvement and future work

As mentioned in Section 5.3, it is essential to have another baseline that uses a more classical methodology and to do some post-processing on the results of the models of this project. This would indeed increase the confidence in the results and the pertinence of the approach. It would also be interesting to explore other dimensionality reduction methods, such as variants

of PCA [56] that remove, deep compression of embeddings [57] or using a dedicated encoding network operating directly on the embeddings map [16]. Finally, it would be possible to use embeddings that are directly trained on a newspaper corpus, rather than using embeddings coming from Wikipedia. This could better adapt to the OCR errors present in the text and create embeddings more linked to the newspapers.

Possible future work includes improving the dhSegment architecture, for example by using deconvolution [5] or by introducing dropout cells [58] to improve the generalization of the models. Moreover, it would be relevant to explore other architectures such as ones using region proposal networks [59] to see if they also benefit from textual features. It would also be important to assess if the improvements of textual embeddings map are only affecting newspapers or if they are useful to other types of documents. Finally, it would be interesting to go closer to a true end-to-end network and try to remove the need of external OCR by using a custom network architecture that is able to directly parse the text and use it for its predictions. These potential further studies would give a better understanding of the effect and potential of mixing textual and visual feature for document semantic segmentation.

A Evaluation interface

In order to facilitate the development and to better assess the performances of the different models, a web interface was developed. This appendix will present the two most essential components of the interface and their use cases: the results summary tab and the specific results tab.

Results summary tab

This tab shows statistics of the results of each experiment, it is shown in figure [A.1](#). A table at the bottom left allows to choose which experiment to compare. Several buttons and a slider allows to choose different parameter for the evaluation. It possible to choose:

- To plot bars with standard deviation or boxplots
- To show the results averaged or at the class level
- Which metric to compute between mIoU, precision, recall and f1-score
- At which threshold to compute the metric

This was particularly useful during development in order to quickly compare different approaches. It allowed to gain a deeper understanding of the different metrics and the influence of the threshold. Some approaches working very badly with high threshold, whereas other were not particularly affected.

Results specific tab

This tab allows to look at the specific of the prediction of each model, it is shown in figure [A.2](#). All the results along with their IoU metric are reported in the table on the bottom left. This table is sortable, allowing to easily consider only bad results. There is two dropdown menus that allow to select the experiment and the class to observe. There also is a button that allow to quickly switch back to the previously selected experiment, making it easier to actually compare them.

This tab was useful to see where the different models made mistakes and if the mistakes were due to an unlucky training or to the model itself. For example if the mistake is present in all the different trainings of the model, then it can be put on the model. This tab was also used to detect mistakes that were made in the annotations, it helped detect not less than forty errors.

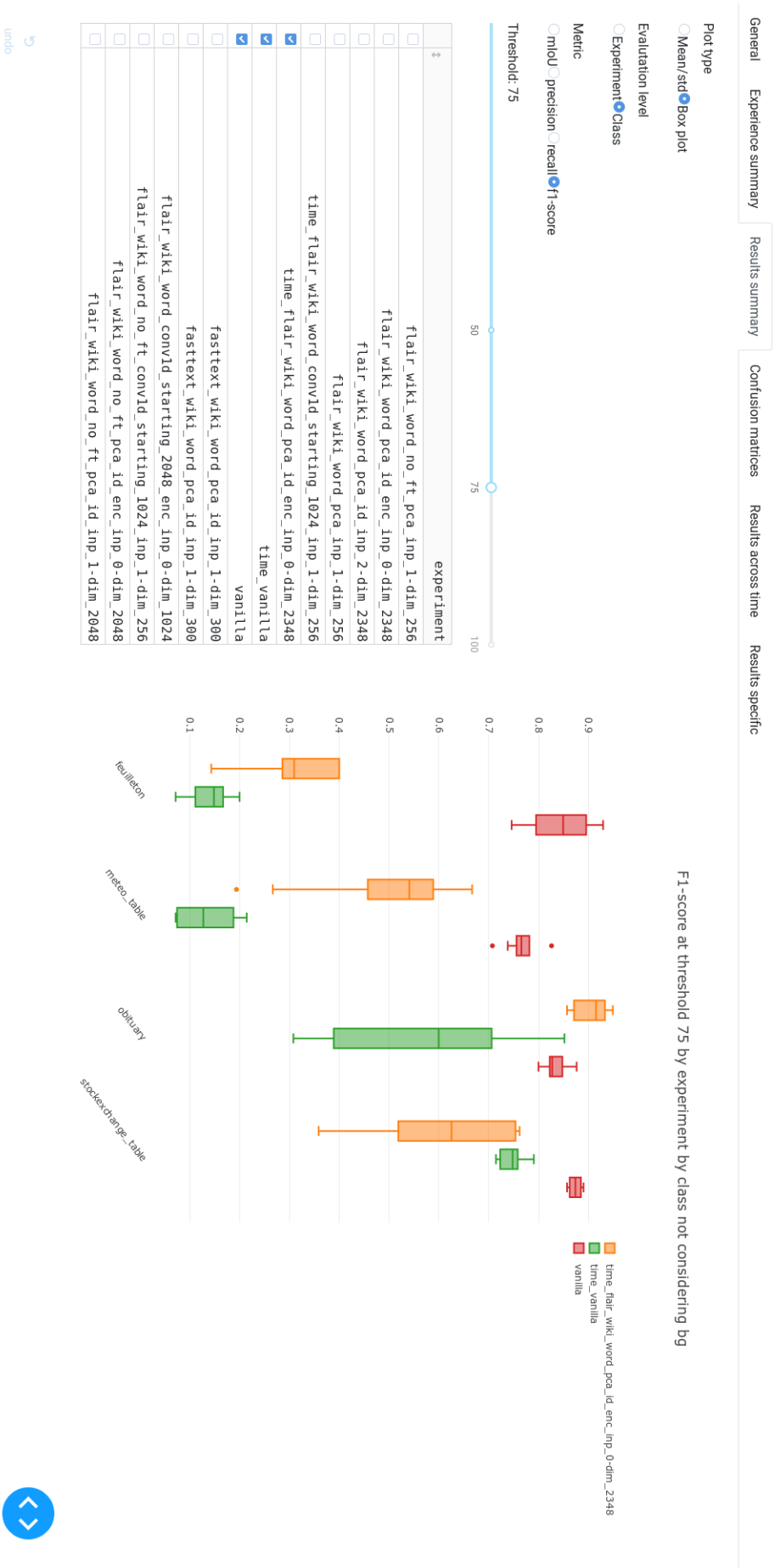


FIGURE A.1: Results summary tab of the interface. It shows a comparison of the F1-scores at a threshold of 75 between the models trained for the time generalization experiment.

General

Experiment summary

Results summary

Confusion matrices

Results across time

Results specific

Experiment

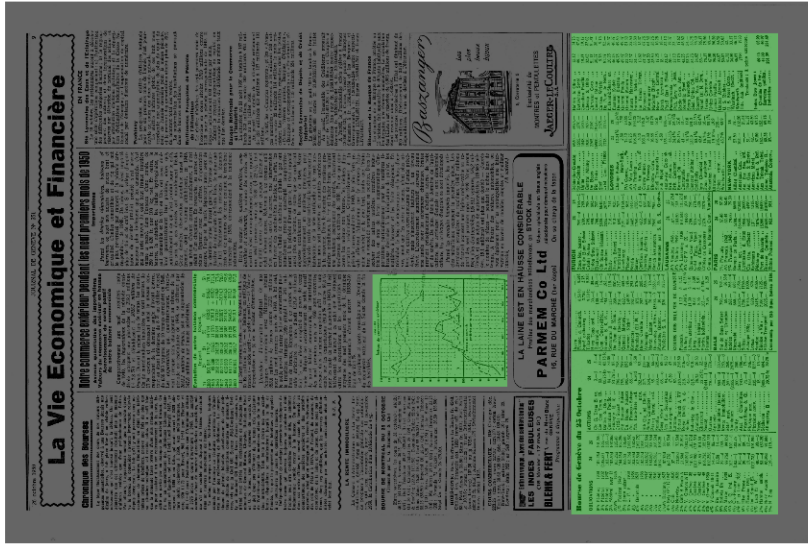
JDG_vanilla

Class

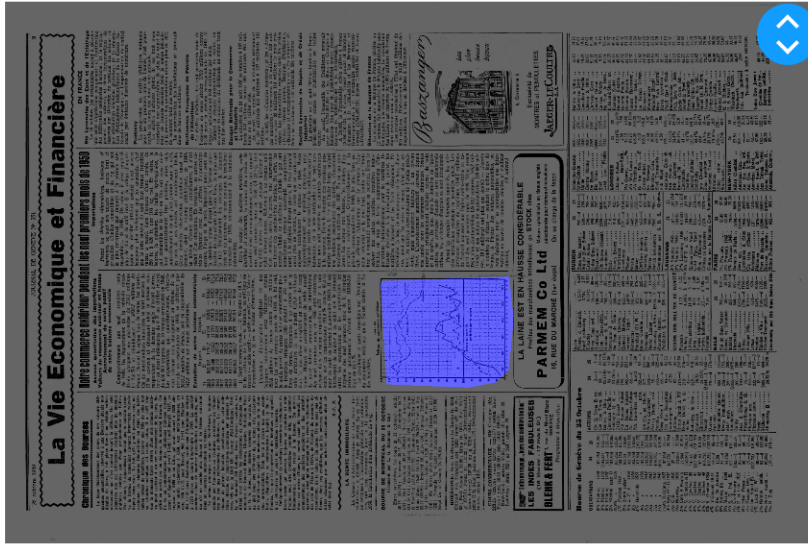
meteo_table

Toggle experiment

Ground-truth



Prediction



	base	name	method	TP	TN	FP	FN	iou
<input type="radio"/>	JDG-1933-04-11-a-	p0008		0	0	0	1	-1
<input type="radio"/>	JDG-1980-12-01-a-	p0027		0	0	0	1	-1
<input type="radio"/>	JDG-1935-04-10-a-	p0008		0	0	0	1	-1
<input type="radio"/>	JDG-1960-04-04-a-	p0010		0	0	0	1	-1
<input type="radio"/>	JDG-1960-04-19-a-	p0012		0	0	0	1	-1
<input type="radio"/>	JDG-1958-12-22-a-	p0010		0	0	0	1	-1
<input type="radio"/>	JDG-1940-02-07-a-	p0003		0	0	0	1	-1
<input type="radio"/>	JDG-1960-05-04-a-	p0014		0	0	0	1	-1
<input type="radio"/>	JDG-1975-04-14-a-	p0007		0	0	0	1	-1
<input checked="" type="radio"/>	JDG-1950-10-26-a-	p0009		0	0	1	0	0
<input type="radio"/>	JDG-1925-05-15-a-	p0006		0	0	1	0	0.31
<input type="radio"/>	JDG-1907-01-23-a-	p0003		0	0	1	0	0.6
<input type="radio"/>	JDG-1882-11-16-a-	p0003		1	0	0	0	0.63
<input type="radio"/>	JDG-1897-12-22-a-	p0003		1	0	0	0	0.7
<input type="radio"/>	JDG-1897-12-11-a-	p0003		1	0	0	0	0.72
<input type="radio"/>	JDG-1867-02-23-a-	p0004		1	0	0	0	0.75
<input type="radio"/>	JDG-1902-05-10-a-	p0003		1	0	0	0	0.76
<input type="radio"/>	JDG-1890-02-22-a-	p0003		1	0	0	0	0.76
<input type="radio"/>	JDG-1930-03-19-a-	p0010		1	0	0	0	0.78
<input checked="" type="radio"/>	JDG-1930-10-08-a-	p0007		1	0	0	0	0.8
<input type="radio"/>	JDG-1933-11-06-a-	p0006		1	0	0	0	0.8
<input type="radio"/>	JDG-1882-04-25-a-	p0004		1	0	0	0	0.81

FIGURE A.2: Results specific tab of the interface. It shows an example where the network mistook a stock exchange table for a weather forecast.

Bibliography

- [1] O. Ronneberger, P. Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* (2015). arXiv: 1505.04597 [cs.CV]. URL: <http://arxiv.org/abs/1505.04597v1>.
- [2] A. Antonacopoulos et al. "ICDAR 2013 competition on historical newspaper layout analysis (HNLA 2013)". In: *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*. IEEE, Aug. 2013, pp. 1454–1458. ISBN: 978-0-7695-4999-6. DOI: 10.1109/ICDAR.2013.293. URL: <http://ieeexplore.ieee.org/document/6628854/>.
- [3] K. Y. Wong, R. G. Casey, and F. M. Wahl. "Document Analysis System". In: *IBM Journal of Research and Development* 26.6 (Nov. 1982), pp. 647–656. ISSN: 0018-8646. DOI: 10.1147/rd.266.0647.
- [4] M. Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.
- [5] J. Long, E. Shelhamer, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015. DOI: 10.1109/cvpr.2015.7298965. URL: <https://doi.org/10.1109/cvpr.2015.7298965>.
- [6] H. Zhao et al. "Pyramid scene parsing network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2881–2890.
- [7] L.-C. Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 801–818.
- [8] M. Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88.2 (June 2010), pp. 303–338.
- [9] T.-Y. Lin et al. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755.
- [10] S. Eskenazi, P. Gomez-Krämer, and J.-M. Ogier. "A comprehensive survey of mostly textual document segmentation algorithms since 2008". In: *Pattern Recognition* 64 (Apr. 2017), pp. 1–14. DOI: 10.1016/j.patcog.2016.10.023. URL: <https://hal.archives-ouvertes.fr/hal-01388088>.
- [11] G. Nagy, S. Seth, and M. Viswanathan. "A prototype document image analysis system for technical journals". In: *Computer* 25.7 (July 1992), pp. 10–22. ISSN: 0018-9162. DOI: 10.1109/2.144436.
- [12] N. Vincent and J. M. Ogier. "Shall deep learning be the mandatory future of document analysis problems?" In: *Pattern Recognition* 86 (2019), pp. 281–289. ISSN: 00313203. DOI: 10.1016/j.patcog.2018.09.010. URL: <https://doi.org/10.1016/j.patcog.2018.09.010>.
- [13] C. Clausner, A. Antonacopoulos, and S. Pletschacher. "ICDAR2017 Competition on Recognition of Documents with Complex Layouts - RDCL2017". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. Nov. 2017, pp. 1404–1410. DOI: 10.1109/ICDAR.2017.229.

- [14] L. Gao et al. "ICDAR2017 Competition on Page Object Detection". In: *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR 1* (2018), pp. 1417–1422. ISSN: 15205363. DOI: [10.1109/ICDAR.2017.231](https://doi.org/10.1109/ICDAR.2017.231).
- [15] R. B. Palm, F. Laws, and O. Winther. "Attend, Copy, Parse - End-To-End Information Extraction From Documents". In: *CoRR* (2018). arXiv: [1812.07248](https://arxiv.org/abs/1812.07248) [cs.CL]. URL: <http://arxiv.org/abs/1812.07248v1>.
- [16] A. R. Katti et al. "Chargrid: Towards Understanding 2d Documents". In: *CoRR* (2018). arXiv: [1809.08799](https://arxiv.org/abs/1809.08799) [cs.CL]. URL: <http://arxiv.org/abs/1809.08799v1>.
- [17] B. Gatos et al. "Integrated algorithms for newspaper page decomposition and article tracking". In: *Proceedings of the Fifth International Conference on Document Analysis and Recognition, ICDAR'99* (Cat. No. PR00318). IEEE. 1999, pp. 559–562.
- [18] K. Hadjar and R. Ingold. "Arabic Newspaper Page Segmentation." In: *ICDAR*. Vol. 3. 2003, pp. 895–899.
- [19] K. Hadjar, O. Hitz, and R. Ingold. "Newspaper page decomposition using a split and merge approach". In: *Proceedings of Sixth International Conference on Document Analysis and Recognition*. IEEE. 2001, pp. 1186–1189.
- [20] B. Meier et al. "Fully convolutional neural networks for newspaper article segmentation". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE. 2017, pp. 414–419.
- [21] W. S. McCulloch and W. Pitts. "A logical calculus of the ideas immanent in nervous activity. 1943." In: *Bulletin of mathematical biology* 52 1-2 (1943), pp. 99–115, 99–115.
- [22] F. Rosenblatt. "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Psychological Review* (1958), pp. 65–386.
- [23] G. Cybenko. "Approximation by superpositions of a sigmoidal function". In: *MCSS 2* (1989), pp. 303–314.
- [24] Y. LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.
- [25] O. Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: [10.1007/s11263-015-0816-y](https://doi.org/10.1007/s11263-015-0816-y).
- [26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Neural Information Processing Systems 25* (Jan. 2012). DOI: [10.1145/3065386](https://doi.org/10.1145/3065386).
- [27] T. Schlegl, J. Ofner, and G. Langs. "Unsupervised pre-training across image domains improves lung tissue classification". In: *International MICCAI Workshop on Medical Computer Vision*. Springer. 2014, pp. 82–93.
- [28] Z. Cao et al. "Realtime multi-person 2d pose estimation using part affinity fields". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7291–7299.
- [29] S. Ares Oliveira, B. Seguin, and F. Kaplan. "dhSegment: A Generic Deep-Learning Approach for Document Segmentation". In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. Aug. 2018, pp. 7–12. DOI: [10.1109/ICFHR-2018.2018.00011](https://doi.org/10.1109/ICFHR-2018.2018.00011).
- [30] K. He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: [1512.03385](https://arxiv.org/abs/1512.03385). URL: <http://arxiv.org/abs/1512.03385>.
- [31] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).
- [32] T. Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. 2013. URL: <http://arxiv.org/abs/1301.3781>.

- [33] Z. S. Harris. "Distributional Structure". In: *WORD* 10.2-3 (1954), pp. 146–162. DOI: [10.1080/00437956.1954.11659520](https://doi.org/10.1080/00437956.1954.11659520). URL: <https://doi.org/10.1080/00437956.1954.11659520>.
- [34] P. Bojanowski et al. "Enriching Word Vectors With Subword Information". In: *CoRR* (2016). arXiv: [1607.04606](https://arxiv.org/abs/1607.04606) [cs.CL]. URL: <http://arxiv.org/abs/1607.04606v2>.
- [35] M. E. Peters et al. "Deep contextualized word representations". In: *Proc. of NAACL*. 2018.
- [36] A. Akbik, D. Blythe, and R. Vollgraf. "Contextual String Embeddings for Sequence Labeling". In: *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. 2018, pp. 1638–1649. URL: <https://aclanthology.info/papers/C18-1139/c18-1139>.
- [37] M. Liao et al. "Textboxes: A fast text detector with a single deep neural network". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.
- [38] P. Mirowski et al. "Learning to navigate in complex environments". In: *arXiv preprint arXiv:1611.03673* (2016).
- [39] V. Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), p. 529.
- [40] B. P. Yuhas, M. H. Goldstein, and T. J. Sejnowski. "Integration of acoustic and visual speech signals using neural networks". In: *IEEE Communications Magazine* 27.11 (1989), pp. 65–71.
- [41] P. K. Atrey et al. "Multimodal fusion for multimedia analysis: a survey". In: *Multimedia systems* 16.6 (2010), pp. 345–379.
- [42] M. Hodosh, P. Young, and J. Hockenmaier. "Framing image description as a ranking task: Data, models and evaluation metrics". In: *Journal of Artificial Intelligence Research* 47 (2013), pp. 853–899.
- [43] T. Baltrušaitis, C. Ahuja, and L.-P. Morency. "Multimodal machine learning: A survey and taxonomy". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.2 (2018), pp. 423–443.
- [44] A. Frome et al. "Devise: A deep visual-semantic embedding model". In: *Advances in neural information processing systems*. 2013, pp. 2121–2129.
- [45] X. Yang et al. "Learning to Extract Semantic Structure from Documents Using Multimodal Fully Convolutional Neural Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017. DOI: [10.1109/cvpr.2017.462](https://doi.org/10.1109/cvpr.2017.462). URL: <https://doi.org/10.1109/cvpr.2017.462>.
- [46] C. Clausner et al. "The ENP image and ground truth dataset of historical newspapers". In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2015, pp. 931–935.
- [47] G. Mühlberger and G. Hackl. *Europeana Newspapers Mets/Alto Profile: D5.3 final public release with updated online resource for documentation*. Tech. rep. Europeana Newspapers, 2015. URL: <http://www.europeana-newspapers.eu/public-materials/deliverables/>.
- [48] A. Dutta, A. Gupta, and A. Zissermann. *VGG Image Annotator (VIA)*. <http://www.robots.ox.ac.uk/~vgg/software/via/>. Version: 2.0.7, Accessed: 14.06.19. 2016.
- [49] G. Lample et al. "Neural architectures for named entity recognition". In: *arXiv preprint arXiv:1603.01360* (2016).
- [50] N. Reimers and I. Gurevych. "Why Comparing Single Performance Scores Does Not Allow to Draw Conclusions About Machine Learning Approaches". In: *arXiv preprint arXiv:1803.09578* (2018).
- [51] G. D. Ruxton. "The unequal variance t-test is an underused alternative to Student's t-test and the Mann-Whitney U test". In: *Behavioral Ecology* 17.4 (2006), pp. 688–690.

- [52] X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.
- [53] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).
- [54] S. Ioffe. "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models". In: *Advances in neural information processing systems*. 2017, pp. 1945–1953.
- [55] V. Buntinx, F. Kaplan, and A. Xanthos. "Layout Analysis on Newspaper Archives". In: *Book of Abstracts of DH2017*. Alliance of Digital Humanities Organizations. Montréal, Canada, Aug. 2017. URL: <https://dh2017.adho.org/abstracts/193/193.pdf>.
- [56] V. Raunak. "Simple and Effective Dimensionality Reduction for Word Embeddings". In: *arXiv preprint arXiv:1708.03629* (2017).
- [57] R. Shu and H. Nakayama. "Compressing word embeddings via deep compositional code learning". In: *arXiv preprint arXiv:1711.01068* (2017).
- [58] N. Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.
- [59] S. Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.