Text/non-text classification of connected components in document images

Frank D. Julca-Aguilar*, Ana L. L. M. Maia*† and Nina S. T. Hirata*
*Department of Computer Science, Institute of Mathematics and Statistics
University of São Paulo (USP)
São Paulo, Brazil
†Department of Exact Science, State University of Feira de Santana (UEFS)
Feira de Santana, Brazil

***Abstract*—Text segmentation is an important problem in document analysis related applications. We address the problem of classifying connected components of a document image as text or non-text. Inspired from previous works in the literature, besides common size and shape related features extracted from the components, we also consider component images, without and with context information, as inputs of the classifiers. Multilayer perceptrons and convolutional neural networks are used to classify the components. High precision and recall is obtained with respect to both text and non-text components.**

I. Introduction

One important process in document image understanding is geometric layout analysis (or page segmentation). It consists in partitioning the image into a set of page components (regions with homogeneous content). Common page components are, for instance, text regions, figures and tables [1]. Another important process is the logical layout analysis which consists in assigning logical labels to the page components based on their contents and relation with other page components [2]. These two processes are often interrelated and there is no clear cutting point between the two in actual solutions [3].

Though correct logical layout analysis is fundamental for a full understanding of document content, detection of only certain types of components is valuable for different purposes. For instance, to build digital catalogs, documents usually do not need to be at a very high resolution, but it is desirable to keep pictures at a higher resolution to preserve their details. This would reduce the amount of data, while preserving the quality of the pictures. Another example is digital document indexing. Since most index entries are based on words found in the documents, text detection and recognition is mandatory while identification of other types of components is not necessarily required. Moreover, once algorithms specialized on detecting particular types of components are available, they can be combined to determine the logical layout of a page.

In this work we address the problem of text region detection in document images. Figure 1 shows a magazine page (binary image) on the left side and the text regions highlighted in blue on the right side. Other non-text page components are shown in yellow.

To approach this problem, one could rely on geometric information such as shapes, sizes, spacing and proximities

Fig. 1. A document image and text components (blue colored regions including the text). Original image from the ICDAR-2009 dataset [4].

between image components, to generate an initial partition of the image. Then, each segment could be classified as text or non-text [1], [3]. It is also possible to start from an over-segmentation of the image (such as from the set of super-pixels or from the connected components) and then perform classification and grouping of these individual segments. Due to variations in page characteristics from document domain to domain, machine learning based approaches constitute a powerful approach to tackle this problem. For segmenting text, classifying connected components is a natural choice in most cases since individual text characters naturally correspond to connected components.

Connected component classification as text or non-text has been recently explored, using machine learning techniques, in [5]–[7]. Shape related information, component images, or a combination of both, are used as features for the classification, using traditional classifier models such as multilayer neural networks and SVM. Component images [5] are image crops centered on a connected component in the image. Hence, they correspond to natural inputs to classifiers of the convolutional neural network (CNN) family. Therefore, in this work we are interested on using CNNs to classify connected components as text or non-text.

Since surrounding information of a connected component may be important to disambiguate identification of objects that resemble text character shapes, we also consider context images, which are component images with surrounding information. Incorporating such information on the feature set in the conventional classification pipeline will require adequate encoding of the surrounding information and it may also imply a non-desired growth in the number of features. On the other hand, for training CNNs, context information can be captured in an image patch with the component at the center. We would like to examine how much classification efficacy is affected by context information.

Most existing ground-truth information on text segmentation is provided in terms of delimiters of the page component regions (usually bounding boxes or enclosing polygons) [4]. Since we are not concerned with delimiting text regions, our analysis is based on connected component classification measures. Conventional learning based classification (the usual method that consists in extracting features and then using them as input in classifier training) serves as a baseline for comparison purposes. We have separated the dataset into training and test sets and used part of the training data to compute validation performance and then choose the best performing model. The chosen model is evaluated on the test set to verify generalization.

Following this, in Section II we review the main related works, and specifically those that use machine learning techniques to classify connected components as text or non-text. Then, in Section III we present an overview of the proposed approach and detail the features and classifiers used in our study. Features are inspired from works described in Section II. We propose some variations on the features and the use of CNNs to classify individual connected component images as well as the context images that include not only the connected component but also the surrounding information. Experiments are performed on the ICDAR-2009 dataset [4]. The proposed method achieves high precision and recall, around 98%, with respect to connected components. At the end, in Section V we present the conclusions of this work.

II. Related works

As stated previously, the aim of this work is to detect text region in document images by means of connected component classification as text or non-text. This work is inspired by the works of Bukhari *et al.* [5] and Le *et al.* [6].

Bukhari *et al.* [5] propose a discriminative learning approach to segment text from document images. They observe that existing methods can be grouped into block classification or pixel classification methods. Block classification methods are those that partition the image into a certain number of blocks and then perform the classification of each of them. They argue that in block classification methods wrong blocks will lead to ill-classification and, on the other hand, although pixel classification methods do not present the same drawback, they are slow. Thus, they propose a method based on connected component classification. To classify a connected component, simple features are extracted from it and fed to a previously trained multi-layer perceptron. Specifically, the features considered are the vectorized forms of the connected component image and surrounding context image, both rescaled to a 40×40 image, plus four size related features, totaling 3204 features.

Le *et al.* [6] follow a similar approach to the one by Bukhari *et al.* [5], but consider a different set of features and classifier model. They do not consider image rescaling, but consider a larger number of features related to size, shape, stroke width and position of the connected components. Their method include a pre-processing and post-processing steps. In the pre-processing step some components are filtered out; in the post-processing step, individual component labels are compared to the labels of the nearest neighbor components and reclassified if they meet a given criterion. A total of seventeen features is used. For classification, a two-class Discrete Adaboost model is used.

While the two works above perform explicit feature extraction, some works rely on convolutional neural networks (CNN) to implicitly extract relevant features. For instance, Rashid *et al.* [8] use CNN to identify Greek and Latin scripts in ancient documents, also by performing connected component classification. They use the same approach of rescaling the component image to 40×40 size images. We observe that CNNs are being increasingly used to process document images. Many of these works are related to document classification and retrieval [9], [10].

III. Proposed approach

In this section we describe the method used to train the classifiers and to select the best model. Application of the method is described in the next section. The method follows the pipeline presented in Fig. 2, which is composed of three main steps: (i) data extraction, (ii) training, and (iii) model selection.

Prior to the first step, all images were binarized using the Otsu's method and then connected components were computed in each image. Data extraction is the process of reshaping the individual connected components of an image to a format suitable to be used by the classifier. For each connected component we extracted a set of features, which we call geometric features, and we also built two images of size 40×40, called as component and context images, respectively, as described below.

A. Training data

The 19 geometric features we extracted are related to size, shape and position information. Most of them have been used by Bonakdar [11] and Le [6]. We add a few others, and thus considered the following features:

- *Height:* Component bounding box height.
- *Width:* Component bounding box width.
- *Aspect Ratio:* Length divided by height.

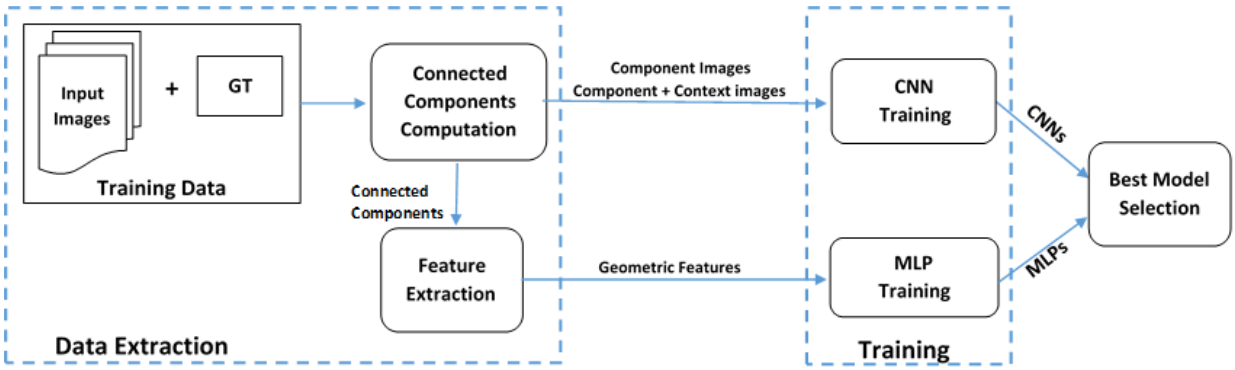


Fig. 2. Training pipeline for connected component classification as text or non-text.

- *Elongation:* Ratio of component height and component width. Computed as $min(height, width)/max(height, width)$.
- *Solidity:* Ratio of component pixels to pixels of the convex hull component image.
- *HU moments [12]:* Scale, rotation and reflection invariant features that describe the shape of connected components. We use only the first four of seven HU moments.
- *X-Y coordinates:* Center of the connected component coordinates.
- *Area:* Number of component pixels.
- *Convex area:* Number of pixels of convex hull component image.
- *Eccentricity:* Ratio of the focal distance (distance between focal points) over the major axis length. The value is in the interval [0, 1).
- *Euler number:* Number of objects (= 1) subtracted by number of holes (8-connectivity).
- *Extent:* Ratio of component pixels to pixels in the total bounding box. Computed as area / (rows * cols)
- *Orientation:* Angle between the X-axis and the major axis of the ellipse that has the same second-moments as the region.
- *Perimeter:* Perimeter of object which approximates the contour as a line through the centers of border pixels using a 4-connectivity.
- *Filled area:* Number of pixels of component image with filled holes.

Component images and context images of connected components were built in a similar way as done in Bukhari *et al.* [5]. There, to build the component images, components whose size was no larger than 40 were placed at the center of a 40×40 image, and those larger than 40 were rescaled so that the largest side were reduced to 40, without changing the aspect ratio, and the rescaled component were placed at the center of the 40×40 image. As for the context images, they also included other components present in the neighborhood of the target component. The size of the neighborhood was computed as a function of the height (h) and the length (l) of the component. Specifically, the region of size $5l$ by $2h$, centered at the component center, is cropped from the image and then rescaled to a 40×40 size image.

We build the component images in the same way as Bukhari *et al.*, obtaining a 40×40 image for each component. As for the context images, we consider a rescaling in such a way as to have all components at a fixed size. A neighborhood large enough to result in a 40×40 image after rescaling the component to the fixed size is cropped around the component. We consider four different fixed sizes for the component (8×8, 12×12, 16×16 and 20×20). Figure 3 shows examples of a component image and the context images for the four sizes. Note that the smaller the rescaled component, more context information is captured. In addition, in the context image we set the component pixels as positive and the pixels in the remaining components as negative. In the figure, white represents the positive pixels, gray represents the negative pixels and black represents the background.

(a) (b) (c) (d) (e)

Fig. 3. Example of component and context images. (a) Component image. Context image with the component within a (b) 8×8, (c) 12×12, (d) 16×16, and (e) 20×20 squares at the center of a 40×40 image.

B. Training and model selection

The data extraction step generates three categories of training data: (a) geometric features, (b) component images, and (c) context images. In the second step of the pipeline, two classifier models are considered. Multilayer Perceptrons (MLP) are trained using the geometric features and Convolutional Neural Networks (CNN) are trained using either component images or context images. All features were normalized.

It is well known that CNN training involves optimizing a large number of parameters. To find an adequate CNN for each input category, rather than varying parameters exhaustively, we first defined a basic architecture (based on a preliminary evaluation) and then we varied a subset of the parameters on

the chosen architecture. The chosen architecture is composed of two sequences of convolution–ReLU–max-pooling layers, followed by two fully connected layers, and an output layer with softmax function. In all training sessions we applied dropout regularization in the penultimate layer, and for cost optimization, we used the Adam algorithm [13]. A grid search of the following parameters have been performed:

- learning rates in $\{1^{-5}, 3^{-5}, 1^{-4}, 3^{-4}\}$;
- convolutional layers in $\{((5,5,1,32),(5,5,32,64)),((3,3,1,32),(3,3,32,64))\}$, where a convolutional layer is defined as a tuple (filter-height, filter-width, #channels, #filters). In all convolutional layers, we used the *same* padding;
- convolution strides in $\{(1,1,1,1),(1,2,2,1)\}$, where one convolution stride is defined as a tuple (batch-stride, height-stride, width-stride, channel-stride).

From the combination of the subset of parameters, the total number of CNN configurations per input category is 16. For each configuration, we trained a CNN with up to 50 epochs, stopping the process when there were no accuracy improvement in the validation set in 5 consecutive epochs (early stopping).

For the Multilayer Perceptron Networks, we defined an architecture with 19 input units (number of geometric features), one output layer with two units with sigmoid function (one for each class), and evaluated architectures with one and two hidden layers with varying number of units. We also tested different values for the learning rate.

The model selection step consists of the following. The trained classifiers were evaluated individually on the validation set. Then, the best one for each input category were selected. The selected CNN and MLP classifiers were combined through a Bayes ensemble and also evaluated. The one with best accuracy on the validation set was chosen.

IV. Experimentation

A. Experimental setup

We used the ICDAR-2009 dataset [4], which contains 55 images. After binarization, two resulted in poor binary images due to the presence of colored text that were not satisfactorily handled by the Otsu algorithm. Instead trying to fix the binarization result, we opted to remove those images from our experiments. Considering the remaining 53 images, we randomly selected 28 images (almost half of the data) for training and 25 images for testing. The $137,862$ components of the training images were randomly divided into validation part (20%) and training part (80%).

Note that during the training and model selection process, evaluation of the classifiers is performed on the components in the validation set and these components may have come from any of the images in the training set. Although this choice may lead to an overestimated validation accuracy, it helps to avoid overestimation or underestimation of the validation accuracy that may result due to the presence of particularly favorable or unfavorable images in the validation set. This concern is also relevant due to the training-test (rather than a cross-validation) scenario we have adopted. Testing was performed on the components that belong to images not-seen during training. We did not apply any pre-processing on the set of extracted components nor any post-processing (for instance, reclassification).

B. Best classifier

Training for each of the parameter variations described in the previous section were executed and individual performance on the validation set were recorded. We also evaluated the performance of the combination of the best classifiers in each category based on Bayes ensemble. However, we did not find improvements over the individual classifiers and therefore they are not reported here.

Table I shows the validation accuracy, recall and precision in terms of text and non-text components, of the best model for each input category. The first row refers to the MLP using the geometric features, the second to CNN using component images, and the last four to CNN using context images, for components rescaled respectively to 20×20, 16×16, 12×12, and 8×8.

TABLE I
PERFORMANCE COMPARISON OF THE METHODS (OVER THE VALIDATION SET)

Method	**Accuracy**	**Non-text**		**Text**	
		recall	**prec.**	**recall**	**prec.**
geom. feat. + MLP	95.91	93.57	91.33	96.76	97.63
20×20 + CNN	98.30	96.66	**97.11**	**98.92**	98.74
20×20 + cont. + CNN	98.56	97.68	97.07	98.89	99.12
16×16 + cont. + CNN	98.53	97.54	97.10	98.90	99.07
12×12 + cont. + CNN	98.45	97.55	96.81	98.79	99.07
8×8 + cont. + CNN	**98.68**	**98.15**	97.06	98.88	**99.30**

Comparing the best classifiers of each category, there is a difference of about 3% between the MLP trained with geometric features and the CNNs using images. Also, we see a consistent improvement, in terms of accuracy, when using context images in comparison to using component images (without context). Among the CNNs, the one trained with context images with 8×8 components presented a slight better accuracy. Thus, we selected that model for the evaluation over the test set.

The selected model presented an accuracy of 97.96% on the test set. With respect to non-text components, the recall was 96.46% and the precision was 96.04%, while with respect to text components, the recall was 98.52% and the precision was 98.68%. These results, very similar to the validation results, indicate that the selected model generalizes well.

Both [5] and [6] also use the ICDAR-2009 in their experimental part. However, the first uses only 8 images and the second reports a five-fold cross-validation result, using for training a subset of the components in the training fold. Since we decided to separate model selection from performance evaluation, we did not perform cross-validation. Although

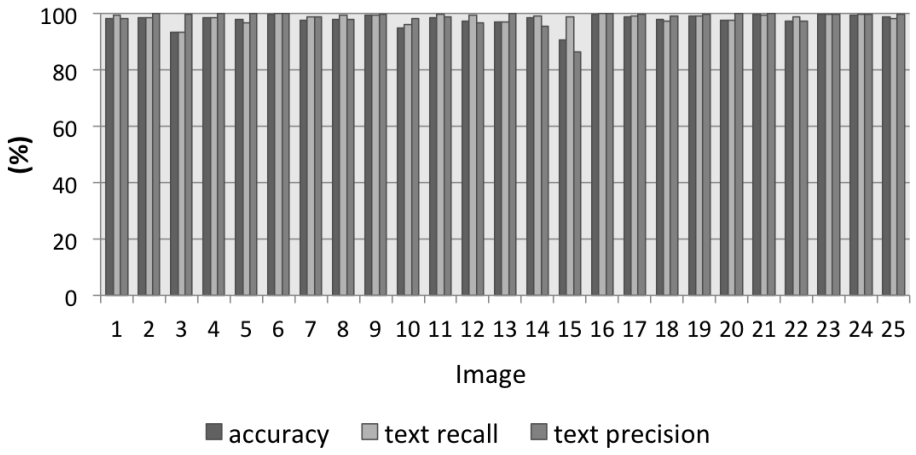


Fig. 4. Accuracy, text recall and precision of the selected model per image in the test set.

we could have done our model selection based on cross-validation over the training set, CNN training is a time consuming process and thus we opted to apply the simple training-validation scheme. Comparing our test results with the cross-validation results reported in [6], w.r.t text component classification our results are similar to theirs but w.r.t. non-text components our results are significantly superior (they report precision of 98.89% and recall of 99.09% w.r.t. text components and precision of 66.72% and recall of 63.35% w.r.t. non-text components).

Figure 4 shows our best model's accuracy, precision and recall with respect to text components per test image. We can see a stable accuracy over the majority of the images. The lowest accuracy was 90.54% w.r.t. image 15 in the figure. This same image presented low text precision (80.0%), meaning that many non-text components were classified as text. As shown in Figure 5, many non-text components around the picture in the center of this image are classified as text (white pixels). Another common error of the CNN classifier are texts located in non-text areas, such as within diagrams, as shown in Figure 6. Although those components are non-text according to ground-truth annotation, it can not be denied that they are texts within a diagram.

V. Conclusion

We have evaluated performance of CNNs to perform text/non-text classification of connected components of document images. We proposed a new context image in which the connected components are rescaled to a given fixed size and the amount of surrounding context is just large enough to fit the rescaled component at the center of a 40×40 image.

We used the results of conventional explicit feature extraction + classification approach as a baseline to evaluate CNN performance. Our conclusion is that CNNs do seem to be able to extract relevant features for the classification, since CNN trained with component images alone presented better results than the MLP trained with the manually extracted features. Moreover, context information, i.e., other components surrounding the component to be classified, seem to have some role in improving performance.

As a future work, we plan to perform a detailed analysis of the types of components that are not correctly classified in order to find ways to improve the results. We also plan to repeat the experiments with CNNs using color images instead of binary images. Additionally, a challenge is to extend this method to documents where text characters are not necessarily isolated components and appear touching other types of objects in the image.

Acknowledgment

F. D. Julca-Aguilar thanks FAPESP (grant 2016/06020-1), A. L. L. M. Maia thanks State University of Feira de Santana, and N. S. T. Hirata thanks CNPq (305055/2015-1). This work was supported by FAPESP (grant 2015/17741-9) and CNPq (grant 484572/2013-0).

References

[1] K. Kise, A. Sato, and M. Iwata, "Segmentation of page images using the area Voronoi diagram," *Computer Vision and Image Understanding*, vol. 70, no. 3, pp. 370 – 382, 1998.

[2] A. Dengel and F. Shafait, "Analysis of the Logical Layout of Documents," in *Handbook of Document Image Processing and Recognition*, D. Doermann and K. Tombre, Eds., 2014, pp. 177–222.

[3] T. A. Tran, I. S. Na, and S. H. Kim, "Page segmentation using minimum homogeneity algorithm and adaptive mathematical morphology," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 19, no. 3, pp. 191–209, 2016.

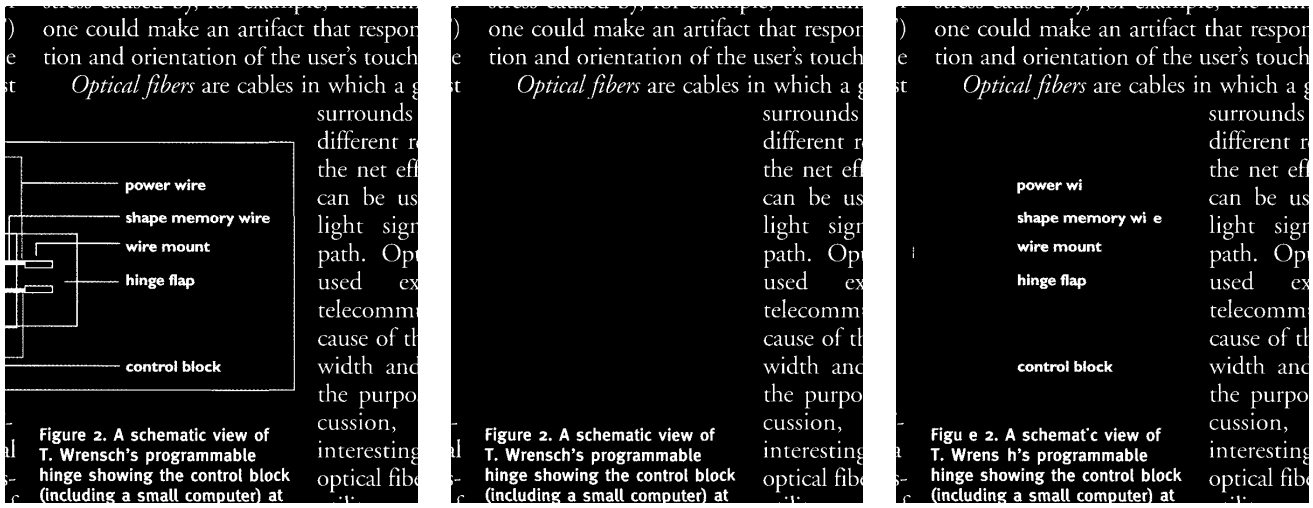
[4] A. Antonacopoulos, S. Pletschacher, D. Bridson, and C. Papadopoulos, "ICDAR 2009 Page Segmentation Competition," in *Proceedings of the 10th International Conference on Document Analysis and Recognition*. IEEE Computer Society, 2009, pp. 1370–1374.

[5] S. S. Bukhari, M. I. A. Al Azawi, F. Shafait, and T. M. Breuel, "Document image segmentation using discriminative learning over connected components," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. ACM, 2010, pp. 183–190.

[6] V. P. Le, N. Nayef, M. Visani, J. M. Ogier, and C. D. Tran, "Text and non-text segmentation based on connected component features," in *International Conference on Document Analysis and Recognition (ICDAR)*, Aug 2015, pp. 1096–1100.

(a) (b) (c)

Fig. 5. Worst result in the test set: (a) input image, (b) expected output, and (c) resulting output.



(a) (b) (c)

Fig. 6. Text within a diagram: (a) input image, (b) expected output, and (c) resulting output (text within the diagram has been recognized as text, contrary to annotation in the ground-truth).

[7] Y. Aramaki, Y. Matsui, T. Yamasaki, and K. Aizawa, "Text detection in manga by combining connected-component-based and region-based classifications," in *IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2901–2905.

[8] S. F. Rashid, F. Shafait, and T. M. Breuel, "Connected component level multiscript identification from ancient document images," in *Proceedings of the 9th IAPR Workshop on Document Analysis System*, 2010, pp. 1–4.

[9] L. Kang, J. Kumar, P. Ye, Y. Li, and D. Doermann, "Convolutional neural networks for document image classification," in *Proceedings of the 22nd International Conference on Pattern Recognition*. IEEE Computer Society, 2014, pp. 3168–3172.

[10] A. W. Harley, A. Ufkes, and K. G. Derpanis, "Evaluation of deep convolutional nets for document image classification and retrieval," in *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR)*. IEEE Computer Society, 2015, pp. 991–995.

[11] O. Bonakdar Sakhi, "Segmentation of heterogeneous document images: an approach based on machine learning, connected components analysis, and texture analysis," Ph.D. dissertation, Paris Est, 2012.

[12] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." *ICLR*, 2014.