# Paragraph text segmentation into lines with Recurrent Neural Networks

Bastien Moysset*§, Christopher Kermorvant†, Christian Wolf‡§, Jérôme Louradour*
*A2iA SA, Paris, France
†Teklia SAS, Paris, France
‡Université de Lyon, CNRS, France
§INSA-Lyon, LIRIS, UMR5205, F-69621

***Abstract*—The detection of text lines, as a first processing step, is critical in all text recognition systems. State-of-the-art methods to locate lines of text are based on handcrafted heuristics fine-tuned by the image processing community's experience. They succeed under certain constraints; for instance the background has to be roughly uniform. We propose to use more "agnostic" Machine Learning-based approaches to address text line location. The main motivation is to be able to process either damaged documents, or flows of documents with a high variety of layouts and other characteristics. A new method is presented in this work, inspired by the latest generation of optical models used for text recognition, namely Recurrent Neural Networks. As these models are sequential, a column of text lines in our application plays here the same role as a line of characters in more traditional text recognition settings. A key advantage of the proposed method over other data-driven approaches is that compiling a training dataset does not require labeling line boundaries: only the number of lines are required for each paragraph. Experimental results show that our approach gives similar or better results than traditional handcrafted approaches, with little engineering efforts and less hyper-parameter tuning.**

## I. Introduction

The detection of text lines is a first processing step in all text recognition systems. The level of accuracy of text boundary locations is critical for the performance of the whole system. In particular, any word missed by the text detection algorithm directly contributes to increase the lower bound of the Word Error Rate that can be expected.

Most of state-of-the-art methods to locate lines of text are based on image processing with handcrafted heuristics fine-tuned by shared experience on digital images. This is the case for the classical projection based methods where lines are separated by finding the minima in the horizontal projection histograms [1] or for techniques using morphological operations or blurring [2], [3] to fuse the letters belonging to the same line. Techniques that try to find minimum paths to join left and right sides of the page [4] or methods extracting text components and grouping them by finding alignment with Hough [5] transform or sweep-line [6] also belong to this category. These techniques usually work well on the specific tasks they were designed for but have difficulties to cope with datasets presenting important variations between the documents or complex backgrounds.

A few techniques have been proposed using machine learning to detect text lines. Delakis et al. [7] use a convolutional neural network and a sliding window to classify each part of a real scene document as text or non-text. Jung [8] present a similar approach and Hebert et al. [9] uses CRF. Although based on learning, these techniques require post-processing based on heuristics to join the positions classified as text. Moreover, they are often applied to printed text only. With handwritten documents, the presence of skew and of overlapping between lines due to ascenders and descenders make the interline gaps less clear and increase the risk for these methods to cause merges between consecutive lines.

In this paper, we propose to use a more "agnostic" approach based on machine learning to address the text line location problem. The main motivation is to be able to process either damaged documents or flows of documents with a high variety in their layouts and other characteristics without too much engineering. A new method based on Deep Neural Networks is presented in Section II. The database and the technical details we use to evaluate this method are given in Section III. Section IV presents the results before drawing some conclusions.

## II. Proposed Text Line Segmentation approach

The proposed method is inspired by the latest generation of optical models used for text recognition, namely Recurrent Neural Networks (RNN) [10], or, to be more specific, one of their more recent variants. These networks are sequential models and therefore take as an input a sequence of observations over a single dimension, which is often time, or the reading direction of a text line in text recognition. The networks produce an output at each step. Traditional (non sequential) neural networks, for instance multi-layer perceptrons, model the mapping from input to output through inter-layer connections defined as linear functions combined with non-linear activation functions. In contrast, recurrent networks add additional recurrent connections between subsequent layers in the sequence, which are able to model the evolution of a signal over "time".

### *A. Sequence-to-sequence learning using multi-directional LSTM-RNN*

In our application, inputs are image patches from a paragraph of a document image. Feature extraction is implicit in this "deep model", pixel values are directly fed into the network. The network outputs predictions for a sequence of line and interline labels, which is essentially a two-class problem. The network is trained using a softmax activation function in the last layer and therefore outputs a sequence of

vectors of posterior probabilities, each vector corresponding to a given patch of the input images.

Unlike RNNs applied to text recognition tasks [10], which scan images horizontally to detect characters, the RNN in our application scans the image vertically to detect horizontal lines of characters. However, in order to integrate information from an entire text line into the decision, we resort to a 2D version of these models with 2D-recurrent connections. To be more precise, each hidden layer is connected to two different preceedings layers, a vertical one and a horizontal one.

We resort to an RNN variant called Long-Short Term Memory (LSTM) [11], which are powerful artificial neurons capable of modelling both local and scattered phenomena within the input images. Four LSTM layers were applied in parallel, one for each possible scanning direction, and their outputs were combined. We trained the RNN using Stochastic Gradient Descent on the posterior probability of the sequence.

One drawback of tackling image segmentation though machine learning is the creation of densely labeled groundtruth, which mostly requires tedious manual annotation. To avoid this situation we only created weak annotations, in particular our method only requires the number of text lines for each paragraph. Given the incomplete groundtruth data, the RNN is trained using Connectionnist Temporal Classification (CTC) [12]. This training procedure efficiently maximises the log-probability over the sum of all possible sequences corresponding to the target sequence. Note that CTC provides our system the freedom to choose how to segment a paragraph into lines.

### B. Problem formulation

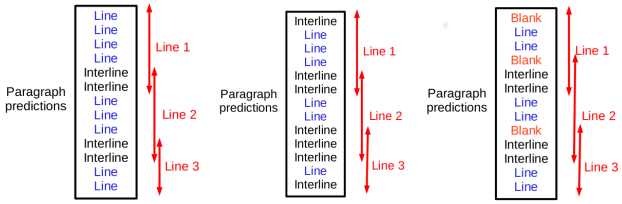
The targets of the training data are computed using the line breaks of the paragraph text in annotations. But there are several ways of designing and training the system:

During the training, the system is given as target a succession of line and interline labels. Recurrent Neural Networks are often used with an added blank output that models the transition between two consecutive elements.

Moreover, the top and the bottom of the paragraphs may look like the interlines. For this reason, if interline labels are used, we can wonder if it is preferable to start or end with an interline target or not, as illustrated in Figure 1 (a) and (b). For this reason, several training data generations are tried and tested in Section IV-A.

Because the neural network is trained with a soft-max activation function in the last layer, it outputs, for each step in the vertical sliding window, a posterior probability for each label, line and interline. For this reason, we can choose the label with the maximum probability at each position. Since training is performed with CTC, the RNNs do not learn the exact line positions during training. They can converge toward different points. It can be the middle of the line or the bottom of it that ends to be detected. The size of the interlines with respect to the size of the lines can also vary. That is why some tuning is needed to define what part of the upper, and respectively lower, interline have to be included in the box.

For the RNN we chose in Section IV-C, boxes are created by fusing the successive line predictions and both side interline




(a) Starting with lines (b) Starting with interlines (c) With blanks

Fig. 1: Columns are considered as vertical sequences of labels. Several possible models have been studied with differences in alphabets and start/end labels. Final line boxes are created from the predictions by joining all consecutive line predictions and surrounding interline or blank predictions.

predictions as illustrated in Figure 1. It means that the zones predicted as interlines will be included at the same time in both the upper and the lower text line boxes. This is particularly interesting for handwritten text because ascenders and descenders are often overlapping.

An illustration of the obtained results for a letter document is given in Figure 2. The red boxes correspond to the paragraph bounding boxes, i.e. to the images that are sent to the optical model. On the left of the red boxes is given the sequence of the predicted labels of the paragraph. A red 'l' for a line prediction and a red 'b' for an interline or a blank prediction. The blue boxes correspond to the predicted line boxes after the post-processing detailed in Figure 1.

## III. Experimental setup

### A. The Maurdor database

Our experiments are made on the documents from the Maurdor database [13]. The Maurdor database is multi-lingual (French, English, Arabic) with both handwritten documents and printed documents. This base is composed of 8774 pages with 192536 zones. The details can be found in Table I. The bounding polygons of the paragraphs are given in the annotations and will be used in this work.
The documents of the Maurdor database are classified in five different categories of documents illustrated in Figure 3. The presence of line breaks in the annotations related to the paragraph texts enabled us to know the number of lines in each paragraph and to build our training data.

### B. Performance metrics

Two metrics are used to assess the efficiency of the line segmentation algorithms.

*1) Error Rate in predicting the right number of lines:* The first one is the percentage of paragraphs in which the right number of lines are detected. It doesn't depend on the position of the lines, just of the number detected. Thus, this metric is not influenced by the line construction described in Section II-B. Meaning that this metric gives fast results without requiring any parameters tuning and avoids the bias of using a post-processing chosen for a particular network. It will be used extensively in the comparison of several versions of our RNNs.

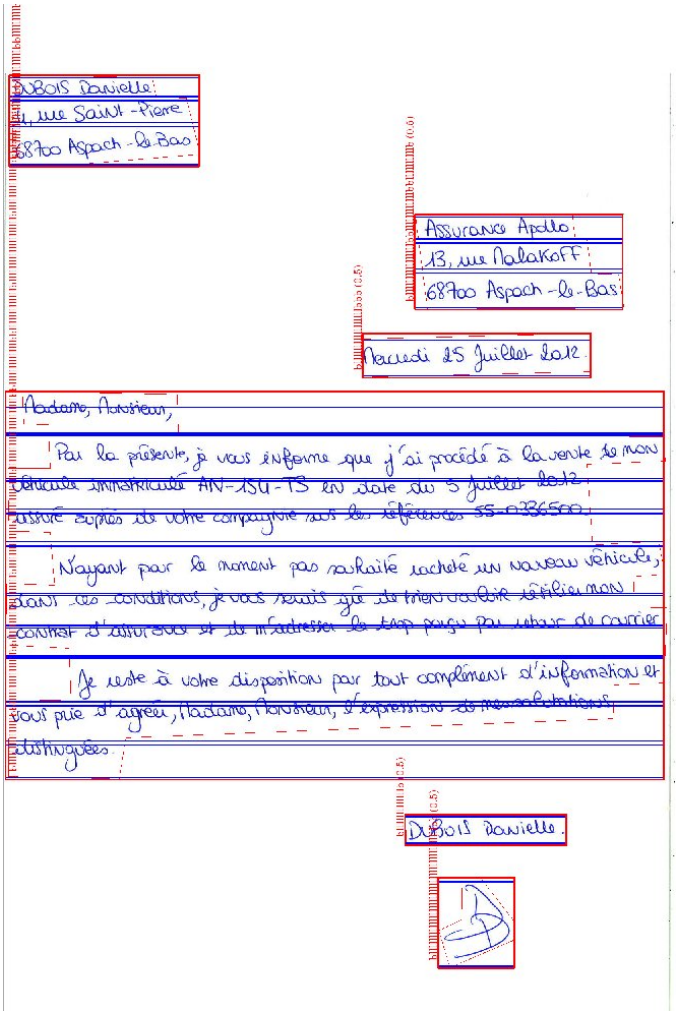


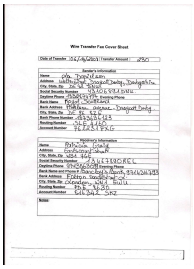

Fig. 2: Illustration of the segmentation obtained for a letter document image

*2) Word Error Rate (WER%):* The second one is the word error rate. The line segmentation algorithm is included in a recognition chain with another recurrent neural network as optical model for text recognition and a language model. The description of the chain can be found in [14]. This metric shows the improvement with respect to the real goal of the line detector algorithm, improving the text recognition. The drawback is that this metric is dependant of a post-processing that may be specific to the RNN. For this reason, extra engineering efforts are necessary. Therefore, this metric will mainly be used on only a few trained networks for comparison with other line segmentation approaches.

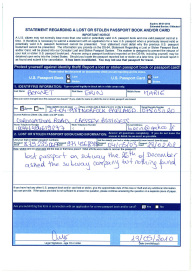
### C. Technical details of the system

*1) Image pre-processing:* Some pre-processing steps are required to homogenize the input paragraph images. These transformations are automatically performed both before training and decoding, they do not need any user intervention.

- Some paragraphs of the Maurdor dataset may be oriented differently than the rest of the page. In such a case, the paragraphs are rotated of 90, 180 or 270 degrees with respect to the data in the annotations.
- The paragraph is rescaled to 300 dpi resolution.
- Our method enabling just to place limits between two successive lines, skewed paragraphs may be a problem. Thus, a deskewing is performed with Bloomberg's algorithm [15].
- For the paragraphs written in Arabic, a horizontal flip of the image is performed to get the alignment of the lines on the left of the images and to get a skew more comparable to those of paragraphs written in Latin. This normalization is useful when networks are trained

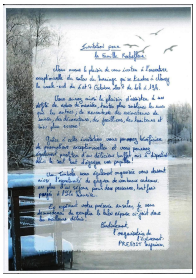


C1: Printed forms with handwritten information



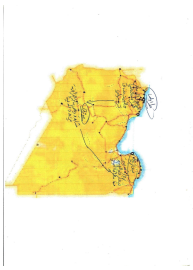
C1: Printed forms with handwritten information

C2: Commercial documents



C3: Handwritten private correspondences


C4: Printed and handwritten private or professional correspondences



C5: Other kinds of documents

Fig. 3: Samples of documents from the Maurdor database with the given category.

TABLE I: The Maurdor database: official splits in Train, Dev and Test sets with the number of text zones for each writing types and languages

| Set | Pages | Zones | | | | | |
|---|---|---|---|---|---|---|---|
| | | Printed zones | | | Handwritten zones | | |
| | | French | English | Arabic | French | English | Arabic |
| Training | 6 592 | 57 821 | 105 002<br>25 773 | 21 263 | 18 417 | 36 681<br>8 530 | 9 729 |
| | | | | 141 683 | | | |
| Validation | 1 110 | 9 908 | 19 205<br>5 124 | 4 122 | 2 857 | 6 458<br>1 765 | 1 835 |
| | | | | 25 663 | | | |
| Test | 1 072 | 11 519 | 18 907<br>4 131 | 3 210 | 3 241 | 6 273<br>1 450 | 1 582 |
| | | | | 25 180 | | | |
| Total | 8 774 | 79 248 | 143 114<br>35 028 | 28 595 | 24 515 | 49 412<br>11 745 | 13 146 |
| | | | | 192 526 | | | |

on both Arabic and Latin paragraphs.

- Finally, paragraph width normalisation is performed. A paragraph width of 200 pixels is found to be a good compromise between performance and processing time.

TABLE II: Number of hidden (and output) units per layer, used in the Recurrent Neural Networks.

| Layer: | Filter size | Number of hidden units | Number of free parameters |
|---|---|---|---|
| (1) LSTM | | 2 | 360 |
| (2) Convolution | 2x4 | 6 | 384 |
| (3) LSTM | | 10 | 5400 |
| (4) Convolution | 2x4 | 20 | 6400 |
| (5) LSTM | | 50 | 121000 |
| (6) Linear | | 3 | 603 |

*2) Recurrent Neural Network architecture:* A six-layer deep neural network is used in this work. It contains three 4-directional LSTM layers and two convolutional layers. Hyperbolic tangent function is used as non-linearity and dropout is added with a probability of 0.5 after the last LSTM layer. The final layer is a fully-connected layer followed by a collapse and a softmax. The size of the convolutional filters, the order of the layers and the number of hidden units on each layer can be found in Table II. The network contains a total of 134147 free parameters.

## IV. Experimental results

Several experiments are done to understand the behaviour of the system and its performance, and to make some design choices. Thus, the results in this part will be given for the Maurdor validation set. When the chosen metric is WER, results will be shown on French sets (both handwritten and printed).

### A. How to model the interlines?

During the training, the system is given as target a succession of line and interline labels. An experiment, presented in Table III, compares the results of a system trained only with two labels "line" and "interline" and a system with a blank label added. We observe that using blank is slightly improving the results.

TABLE III: Word error rates of different blank strategies on Maurdor validation set.

| System | French Handwritten | French Printed |
|---|---|---|
| RNN without blank | 21.43% | 8.01% |
| RNN with blank | 19.45% | 7.58% |

The top and the bottom of the paragraphs may look like the interlines. Bringing in the question of starting (respectively, ending) with a line or with an interline in the target sequences. Table IV shows the performances of both these approaches showing that the system is better when the target sequences seen during the training are starting and ending with a line label. We conjecture that the better performance with starting labels set to line stem from the freedom the network has in its interpretation of the top and bottom blanks. In this case, during training the interline class is not disturbed with top and bottom samples which may have different appearances.

TABLE IV: Word error rates for different labelling strategies on Maurdor validation set.

| System | French Handwritten | French Printed |
|---|---|---|
| Targets starting with line | 19.45% | 7.58% |
| Targets starting with interline | 29.15% | 19.21% |

### B. Do we need to specialize the network?

The results shown in the previous section are related to RNN trained on the full Maurdor training set, namely, the RNN trained at the same time on the two writing types, the three languages, the five document categories. The networks in this section are trained with a blank class and starting labels are set to "line". We tried to train specialized networks on parts of the dataset.

In this section results on the Maurdor validation set will be given as the percentage of paragraphs with the correct number of lines detected in order to avoid the need to tune the way to obtain the line boxes for each RNN. Three attempts of specialisation are performed. Respectively datasets with documents of a given writing type, a given writing script or representing a given category of documents are selected.

In Table V is shown the comparison of networks trained on one writing type only with a network trained on both printed and handwritten paragraphs. In Table VI, we assessed the improvement of training the network on a specific script instead of training it on both Arabic and Latin (French and English) scripts. Finally, Table VII measures specification influence with respect to the different categories described in Figure 3.

The type-specialisation enables to improve the performance of up to 40% for the printed paragraphs, but cause a degradation of 9% for the handwritten paragraphs. The script specialisation in general decreases slightly the error rates, handwritten Latin being the exception with a slight degradation. And most of the specialisations with respect to the document category are giving slightly better performances.

Most of the specialisations bring a slight improvement. For some, a deterioration is observed. The differences between the specialisations can be explained by two opposite effects. Selecting a part of the subset means that the diversity within this set is reduced and that the task will be easier for the neural network. but it also means that the network will see less different samples during the training.

### C. Experiments: comparison with state of the art methods.

In this section, a non-specialized RNN is selected and some post-processing (cf. Section II-B) is selected to optimise its performances on the Maurdor validation dataset. The boxes predicted by the line segmentation algorithm are then sent to a recognition module, described in [14]. Evaluation is done using the WER metric described in Section III-B.

For comparison, the same process is done with other line segmentation algorithms.

- An adaptation of Shi et al. [2] that blurs the image using steerable filters and extract line components from its binarization.

TABLE VIII: Comparison between the word error rates obtained with several line segmentation algorithms on Maurdor test set.

| Line detector | Handwritten | | | Printed | | |
|---|---|---|---|---|---|---|
| | French | English | Arabic | French | English | Arabic |
| RNN - Our work | 25.20% | 36.01% | 31.04% | **9.44%** | **8.81%** | **18.88%** |
| Adaptation of Nicolaou et al. [4] | 27.04% | 41.19% | 34.45% | 11.49% | 11.49% | 28.66% |
| Adaptation of Shi et al. [2] | 37.16% | 44.88% | 40.01% | 27.85% | 39.59% | 31.35% |
| Maurdor campaign - Best single line detector [14] | 25.6% | 42.7% | 33.3% | 18.2% | 15.9% | 22.2% |
| Maurdor campaign - With line segmentation alternatives [14] | **22.2%** | **35.2%** | **29.8%** | 11.3% | 12.8% | 22.8% |

TABLE V: Comparison between type-specialized and type-generic RNNs. Results are given as percentage of paragraphs with the correct number of lines detected.

| Dataset | Specialized RNN | Generic RNN |
|---|---|---|
| Handwritten | 4.20% | 3.83% |
| Printed | 1.50% | 2.58% |

TABLE VI: Comparison between script-specialized and script-generic RNNs. Results are given as percentage of paragraphs with the correct number of lines detected.

| Dataset | Specialized RNN | Generic RNN |
|---|---|---|
| Handwritten Arabic | 3.16% | 3.60% |
| Handwritten Latin | 4.09% | 3.92% |
| Printed Arabic | 1.21% | 2.23% |
| Printed Latin | 2.58% | 2.67% |

TABLE VII: Comparison between category-specialized and category-generic RNNs. Results are given as percentage of paragraphs with the correct number of lines detected.

| Dataset | Specialized RNN | Generic RNN |
|---|---|---|
| Category 1 | 0.95% | 1.26% |
| Category 2 | 3.69% | 4.45% |
| Category 3 | 8.76% | 10.18% |
| Category 4 | 2.93% | 2.24% |
| Category 5 | 2.70% | 2.99% |

- An adaptation of Nicolaou et al. [4] that uses local minima to follow the valleys between the text lines.
- The best single detector for each subset of our system during the Maurdor competition [14].
- A combination of line detectors that is described in [14]

Results on the Maurdor test set can be found in Table VIII. We can see that our approach using RNN for line segmentation beats all the other single line detectors tested for the three languages (French, English and Arabic) and the two writing types (Handwritten and Printed). It gives better results than the combination of line segmentations used in [14] for the printed paragraphs but still slightly worse results for the handwritten zones. Note that the combination performs the recognition several times while our approach just requires to launch it once, enabling consequent processing time reduction.

## V. Conclusion

We have introduced a novel method to segment paragraph text lines using recurrent neural networks that does not need to label the boundaries of the lines and that can work on a high variety of documents without heavy engineering efforts. This technique has shown state of the art performances on the challenging Maurdor database.

Future work could include a combination of this technique with other line segmentation methods, automatic definition of the way to compute reliable line boundaries and attempts to get a more two dimensional approach for full page processing.

## Acknowledgement

This work was partly funded by the French Grand Emprunt-Investissements d'Avenir program through the PACTE project.

## References

[1] A. Zahour, L. Likforman-Sulem, W. Boussellaa, and B. Taconet, "Text Line Segmentation of Historical Arabic Documents," in *International Conference on Document Analysis and Recognition*, 2007.

[2] Z. Shi, S. Setlur, and V. Govindaraju, "A Steerable Directional Local Profile Technique for Extraction of Handwritten Arabic Text Lines," in *International Conference on Document Analysis and Recognition*, 2009.

[3] V. Papavassiliou, V. Katsouros, and G. Carayannis, "A Morphological Approach for Text-Line Segmentation in Handwritten Documents," in *International Conference on Frontiers in Handwriting Recognition*, 2010.

[4] A. Nicolaou and B. Gatos, "Handwritten Text Line Segmentation by Shredding Text into its Lines," *International Conference on Document Analysis and Recognition*, 2009.

[5] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line and word segmentation of handwritten documents," *Pattern Recognition*, vol. 42, no. 12, pp. 3169–3183, Dec. 2009.

[6] I. Rabaev, O. Biller, J. El-Sana, K. Kedem, and I. Dinstein, "Text line detection in corrupted and damaged historical manuscripts," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 812–816.

[7] M. Delakis and C. Garcia, "text detection with convolutional neural networks." in *VISAPP (2)*, 2008, pp. 290–294.

[8] K. Jung, "Neural network-based text location in color images," *Pattern Recognition Letters*, vol. 22, no. 14, pp. 1503–1515, 2001.

[9] D. Hebert, T. Paquet, and S. Nicolas, "Continuous crf with multi-scale quantization feature functions application to structure extraction in old newspaper," in *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE, 2011, pp. 493–497.

[10] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," 2008.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning*, 2006, pp. 369–376.

[13] S. Brunessaux, P. Giroux, B. Grilheres, M. Manta, M. Bodin, K. Choukri, O. Galibert, and J. Kahn, "The maurdor project - improving automatic processing of digital documents," 2014.

[14] B. Moysset, T. Bluche, M. Knibbe, M. F. Benzeghiba, R. Messina, J. Louradour, and C. Kermorvant, "The A2iA Multi-lingual Text Recognition System at the Maurdor Evaluation," in *International Conference on Frontiers in Handwriting Recognition*, 2014.

[15] D. S. Bloomberg, G. E. Kopec, and L. Dasari, "Measuring document image skew and orientation," in *IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*. International Society for Optics and Photonics, 1995, pp. 302–316.