CNN-BLSTM-CRF Network for Semantic Labeling of Students' Online Handwritten Assignments

Amirali Darvishzadeh, Thomas F. Stahovich, Amir Feghahati, Negin Entezari,
Shaghayegh Gharghabi, Reed Kanemaru, Christian Shelton
University of California, Riverside
Email: darvisha@cs.ucr.edu, stahov@engr.ucr.edu, {sfegh001@, nente001, sghar003, rkane005}@ucr.edu, cshelton@cs.ucr.edu

***Abstract*—Automatic semantic labeling of strokes in online handwritten documents is a crucial task for many applications such as diagram interpretation, text recognition, and search. We formulate this task as a stroke classification problem in which each stroke is classified as a cross-out, free body diagram, or text. Separating free body diagram and text in this work is different than the traditional text/non-text separation problem because these two classes contain both text and graphics. The text class includes textual notes, mathematical symbols/equations, and graphics such as arrows that connect other elements. The free body diagram class also contains graphics and various alphanumeric characters and symbols that mark or explain the graphical objects. In this work, we present a novel deep neural network model for classification of strokes in online handwritten documents. There are two input sequences to the network. The first sequence contains the trajectories of the pen strokes while the second contains features of the strokes. Each of these sequences is fed to its own CNN-BLSTM channel to extract features and encode relationships between nearby strokes. The output of the two channels is concatenated and used as the input to a CRF layer that predicts the best sequence of labels for given input sequences. We evaluated our model on a dataset of $1,060$ pages written by 132 students in an undergraduate statics course. Our model achieved an overall classification accuracy of 94.70% on this dataset.**

***Keywords*-Semantic Labeling; Stroke Classification; CNN; LSTM; CRF**

I. Introduction

Semantic labeling of online documents is a crucial prerequisite in document analysis and understanding tasks such as retrieval, recognition, and beautification. In online handwritten documents, semantic labeling is the task of classifying pen strokes into meaningful classes. A pen stroke is a series of time-stamped coordinates beginning when the pen touches the page and ending when the pen leaves the page. Researchers have developed various domain-dependent classification algorithms for specific types of data and specific tasks [1]. Additionally, general methods have also been developed for classification of unconstrained documents [2]–[4]. However, these methods do not produce satisfactory accuracy for some problems. Therefore, classification of pen strokes in unconstrained documents remains a challenging problem that requires more attention from researchers.

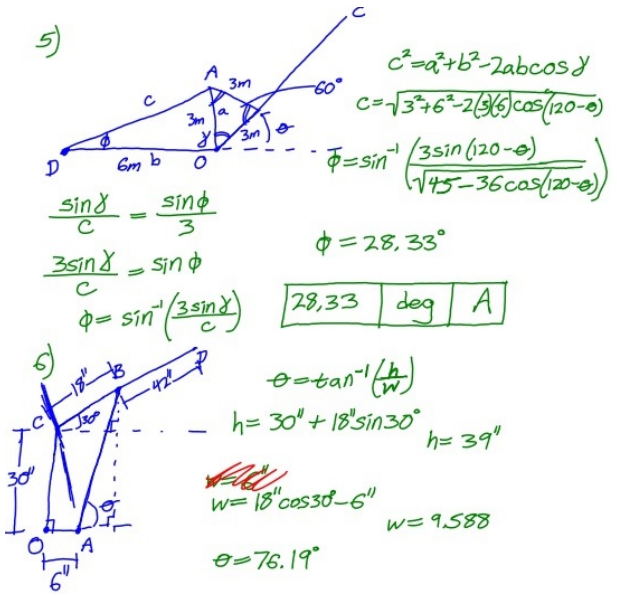

Figure 1: **A typical answer to a free response question in statics course contains a mixture of free body diagrams, equations, text and cross-outs. Free body diagram, cross-out and text classes are shown in blue, red and green, respectively.**

In this paper, we are interested in the task of classifying strokes from students' handwritten homework assignments. More specifically, our dataset comprises handwritten free response solutions to statics problems. Statics is the branch of mechanics that examines the equilibrium of bodies subjected to forces. Participants were students enrolled in an undergraduate statics course. The students used Livescribe smartpens throughout the quarter to take lecture notes and answer questions on exams and homework. The smartpens are used with special dot-patterned paper. A camera at the tip of the pen uses the dots to digitize the writing. Each handwritten page is recorded as a series of strokes that are represented as a sequence of time-stamped coordinates. Our semantic labeling task is to classify strokes into the following three classes, which are illustrated in Figure 1:

- **Free body diagram (FBD):** Diagrams used to represent the forces acting on a mechanical systems. Free body diagrams include drawings, text labels and arrows.
- **Cross-out:** A cross-out is a set of strokes used to cross-out writing. For example, an "X" is a common cross-out mark.
- **Text:** This class comprises all writing that is not free body diagrams or cross-outs. This category primarily comprises equations, but also includes explanatory notes, organizational information (e.g. problem number, student name, etc.), and lines and arrows showing relationships between other writing.

In this work, we introduce a novel deep neural network model for classifying strokes into these three classes. This model employs two Convolutional Neural Networks (CNNs), one that extracts features from pen trajectories and another that extracts domain-dependent stroke features. The output of each CNN is fed into a Bidirectional Long Short Term Memory (BLSTM) network to encode the information between sequences of strokes. Finally, the outputs of the two BLSTM networks are concatenated and passed to a Conditional Random Field (CRF) layer which assigns the final classifications.

The rest of the paper is organized as follows. In Section II we review related work. We present the details of our model in Section III. We discuss the system workflow and network training in Section IV. In Section V, we describe the dataset and empirical evaluation. Finally, in Section VI, we conclude the paper.

II. Related Work

Classification in online handwritten documents has been the subject of much research. Many algorithms have been proposed for this task. In the literature, this task is sometimes called text/graphics separation, text/non-text division, and so on. Here we review existing approaches to this classification task.

Text/non-text separation: Many approaches in this category combine the local features of strokes with contextual information to classify the strokes into text and non-text classes. Techniques proposed by Bishop et al. [5], Zhou and Liu [6] and Delaye et al. [7], initially perform isolated stroke classification and then use a Hidden Markov Model (HMM), Markov Random Fields (MRF), and a CRF, respectively, to model the interactions between neighboring strokes. The approach by Jun-Yu et al. [8] jointly trains a CRF and a neural network (NN) model to combine contextual information with local features. Our task is different as some of our classes contain both text and non-text.

Text line segmentation: This is the task of splitting text strokes into text lines. Shilman et al. [9] use temporal and spatial features of pen strokes to simultaneously perform classification and segmentation of free-form handwritten notes using a dynamic programming approach. Blanchard and Artieres [10] developed a Probabilistic Feature Grammar system for detecting text lines. However, many approaches in this category are not designed to handle heterogeneous documents containing both text and non-text elements.

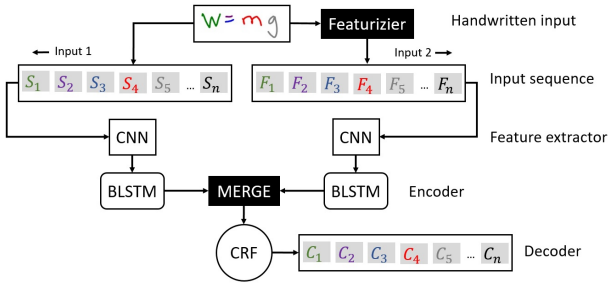

Figure 2: **The overall architecture of CNN-BLSTM-CRF model.**

General segmentation techniques: The approaches in this group present a flexible framework to handle document segmentation for a variety of problems. A first attempt was made by Jain et al. [2]. They build a minimum spanning tree of the non-text strokes and hand-tune the cutting criterion. The system is error-prone because it classifies strokes using only local features. Delaye and Lee [3] construct pairwise distances between strokes and use them with a single linkage clustering strategy to cluster strokes into meaningful objects. To apply the system to new problems requires feature selection and parameter tuning.

III. Proposed CNN-BLSTM-CRF Model for Stroke Classification

In this section, we present our hybrid neural network model for classifying pen strokes from online handwritten documents into three classes: free body diagram, cross-out, and text. Figure 2 shows an overview of the system. In this network, two sets of input sequences are generated from the pen strokes. The first contains the coordinates of the strokes, and the second consists features computed from the strokes. Each of the input sequences is fed to a separate CNN-BLSTM network that extracts new features from the input sequence and encodes the information between stroke sequences. The outputs of the BLSTM layers are concatenated and becomes the input to the CRF layer which decodes this information and generates a probable label for each stroke. Brief description of each layer are provided below.

A. Input sequencing

The input to the network is a page of handwritten pen strokes. Each stroke comprises a sequence of time-stamped coordinates. The i^{th} stroke is represented as

$$S_i = \{[x_1^{(i)}, y_1^{(i)}], ..., [x_m^{(i)}, y_m^{(i)}]\}, \quad (1)$$

where $[x_j^{(i)}, y_j^{(i)}]$ are the coordinates of the j^{th} point. Strokes can have an arbitrary number of trajectory points, but in our

approach, we represent them as having a fixed number of points (m). The parameter m is set to 250 in our experiments to accommodate most of the strokes in our dataset. If a stroke has fewer than m points, zeros are added to the end. If a stroke has more than m points, it is broken down into smaller strokes each have equal numbers of points. The first input sequence is constructed by concatenating the trajectory points of all of the strokes on the page:

$$I_1 = \{[x_1^{(1)}, y_1^{(1)}], ..., [x_m^{(1)}, y_m^{(1)}], ..., [x_1^{(n)}, y_1^{(n)}], ..., [x_m^{(n)}, y_m^{(n)}]\} \quad (2)$$

The value of n is set to the largest number of strokes observed on a page in our dataset. If the page has fewer than n strokes, strokes containing zero coordinates are added to the end of the sequence.

We extract a 16-dimensional feature vector for each stroke and concatenate them together to form the second input sequence:

$$I_2 = \{[f_1^{(1)}, ..., f_{16}^{(1)}], ..., [f_1^{(n)}, ..., f_{16}^{(n)}]\} \quad (3)$$

where $[f_1^{(i)}, ..., f_{16}^{(i)}]$ is the 16-dimensional feature vector corresponding to the i^{th} stroke. Here again if the page has fewer than n strokes, zero features are added to the end of the sequence.

Table I provides an overview of the 16 features, which are taken from [1]. (See [1] for details.) The first 10 features were designed for identifying cross-out strokes. f_{BW} and f_{BH} are the height and width of the minimum bounding box containing the stroke, respectively. f_D characterizes the density of the pen stroke as cross-out strokes may be drawn as a dense "blob". Some cross-out strokes are straight lines; f_{SR} characterizes the straightness of a stroke. Sometimes cross-out comprise sets of parallel strokes or strokes that form an "X". f_p and f_X are boolean-valued features indicating whether a stroke is part of a parallel line, or a cross respectively, with nearby strokes. f_{AO} and f_{AU} are the fraction of the strokes that were drawn earlier or later, respectively. Finally, f_{TU} is the time difference between the stroke and the earliest underlying stroke.

The next six features are designed to distinguish free body diagram from text strokes. Stahovich and Lin [1] observed that the earliest-drawn strokes are more likely to be part of free body diagrams rather than equations. f_{NT} is a normalized time where the earliest-drawn stroke has a normalized time value of 0, and the latest-drawn has a value of 1. f_{LS} has a value of 1 if a stroke is three times taller or wider than the average stroke height. f_{NL} denotes the number of nearby long strokes. Here, two strokes are nearby only if the minimum point-to-point distance between them is less than twice the average stroke height. Strokes written in an equation are typically separated from one another, whereas, many pen strokes on free body diagram intersect each other. This property is captured by f_{IN} which shows the number of intersecting strokes. f_{ID} represents the density of the intersecting strokes and is computed as:

$$f_{ID} = \frac{L^2}{A_{BB}} \quad (4)$$

where L is the sum of the arc lengths of the intersecting strokes and A_{BB} is the area of the bounding box of the stroke. Equation strokes are likely to be drawn from left to right. f_{D2N} is the direction from a stroke to the one drawn next.

As shown in Figure 2, the input sequences I_1 and I_2 are used as the input to the left and right CNN-BLSTM networks, respectively.

Table I: Domain dependant stroke features [1].

Category	Name	Description
Cross-out	f_{BW}	Bounding box width
	f_{BH}	Bounding box height
	f_D	Ink density
	f_{SR}	Straightness ratio
	f_X	Part of a cross?
	f_P	Part of a set of parallel lines?
	f_{AU}	Area fraction under the stroke
	f_{AO}	Area fraction over the stroke
	f_{UH}	Average underlying stroke height
	f_{TU}	Time to first underlying stroke
Miscellaneous	f_{NT}	Normalized time
	f_{LS}	Is long stroke?
	f_{NL}	Number of nearby long strokes
	f_{IN}	Number of intersecting strokes
	f_{ID}	Density of intersecting strokes
	f_{D2N}	Direction to the next stroke

B. CNN (feature extractor)

CNN layers have shown outstanding performance in deriving effective features from images, sequence data, and the like. Each CNN layer performs a linear and a non-linear operation to transform the input data. In this study, CNN layers have one-dimensional kernels which are convolved with the input sequence over a single dimension. Details of the parameter settings are shown in Table II. A CNN network comprises neurons that are arranged in multiple one-dimensional arrays. Each neuron is connected to the neighboring small region of neurons of the previous layer via feed-forward connections. We use a Rectified Linear Unit (ReLU) activation function to apply non-linearity to the neuron values.

C. BLSTM (encoder)

Recurrent neural networks (RNNs) have shown great promise in processing sequential data. Unlike feed-forward networks that pass information in one direction, RNNs are capable of capturing time dynamics through cyclic connections. LSTM networks are a popular and successful type of

RNN that were introduced by Hochreiter and Schmidhuber [11]. Each LSTM unit is equipped with multiple gates that enable it to control the flow of information. Input, output, and forget gates are at the core of an LSTM cell. More specifically, the input gate (i) monitors the incoming data, the forget gate (f) decides what to be discarded from past memory dynamics, and the output gate (o) determines what piece of information to flow out of the cell.

Formally, we use following equations to update an LSTM unit at time step t:

$$\begin{aligned} i_t &= \sigma(W_{h_i} h_{t-1} + U_{x_i} x_t + b_i) \\ f_t &= \sigma(W_{h_f} h_{t-1} + U_{x_f} x_t + b_f) \\ o_t &= \sigma(W_{h_o} h_{t-1} + U_{x_o} x_t + b_o) \\ \tilde{c}_t &= \tanh(W_{h_c} h_{t-1} + U_{x_c} x_t + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{5}$$

where the input vector at time step t is x_t, the hidden state at t is h_t. $U_{x_i}, U_{x_f}, U_{x_o}$, and U_{x_c} are weight matrices for the gates, and $W_{x_i}, W_{x_f}, W_{x_o}$, and W_{x_c} are weight matrices for the hidden state. b_i, b_f, b_o, and b_c are biases for the hidden state. $\tilde{c}_t$ is an intermediate value that is used to update the cell state c_t. Pointwise multiplication operation is denoted by $\odot$. σ and tanh are the sigmoid and hyperbolic tangent functions, respectively.

LSTM units are capable of capturing long-term, past dependencies in their hidden states. Our network computes two sets of contextual information from neighboring strokes. The LSTM network on the right channel of our network (as shown in Figure 2) produces contextual information from domain-dependent features extracted from the pen strokes. The LSTM network on the left channel of our network produces contextual information directly form the pen stroke coordinates.

For many sequence labeling problems, accessing future contextual information enables the model to encode more complex contextual knowledge from the sequential data. As proposed in [12], we use bi-directional LSTM (BLSTM) units in our model. BLSTM layers are composed of two separate LSTM networks where one of them processes the input in the forward direction (i.e., from start to end) and the other operates in the backward direction (i.e., from end to start). The hidden states of the two anti-parallel LSTMs are concatenated together to produce the final output.

D. CRF (decoder)

Many sequence labeling problems require incorporating relations between neighbors in order to predict a label for any individual in a given input sequence. Conditional Random Fields (CRF) are a class of undirected graphical models that are used to compute the conditional probability of output nodes given the values in the input nodes. They are able to jointly predict the best label for each element in the input sequence.

Let $h = \{h_1, ..., h_n\}$ denote an input sequence to the CRF layer which is obtained by concatenating the outputs of the two BLSTMs. More specifically, each h_i comprises the i^{th} hidden state of both the forward and backward LSTMs from both the left and right channels of the model. $y = \{y_1, ..., y_n\}$ is the output sequence of the CRF where each y_i is the predicted label for stroke S_i.

We use a CRF from [13] to model our sequence labeling task:

$$P(y|h;\theta) = \frac{1}{Z(h)} exp\{\phi(h, y; \theta)\} \tag{6}$$

where $\theta = \{W, b\}$ represents the set of parameters, and $Z(h)$ is the normalization factor. $\phi(h, y; \theta)$ is the potential function which is formulated as

$$\phi(h, y; \theta) = \sum_{i=1}^{n} \phi_U(h_i, y_i; \theta) + \sum_{i=1}^{n-1} \phi_P(h_i, h_{i+1}, y_i, y_{i+1}; \theta) \tag{7}$$

where $\phi_U(h_i, y_i; \theta)$ is the unary potential function that measures the compatibility of the i^{th} stroke and the label y_i. $\phi_P(h_i, y_i, y_{i+1}; \theta)$ is the pairwise potential function that captures the dependency between adjacent strokes. These potential functions are formulated as

$$\begin{aligned} \phi_U(h_i, y_i; \theta) &= h_i^T \theta^u_{y_i} \\ \phi_P(h_i, h_{i+1}, y_i, y_{i+1}; \theta) &= h_i^T \theta^{p,1}_{y_i, y_{i+1}} + h_{i+1}^T \theta^{p,2}_{y_i, y_{i+1}} \end{aligned} \tag{8}$$

Here, C is the number of classes. $\theta^u_{y_i}$, $\theta^{p,1}_{y_i, y_{i+1}}$ and $\theta^{p,2}_{y_i, y_{i+1}}$ are the parameters of CRF.

For training purposes, we search for the optimal solution to maximize the conditional likelihood in our training data. We use the logarithm of the likelihood as follows:

$$L(\theta) = \sum_{i=1}^{N} \log p(y|h;\theta) \tag{9}$$

Here, N is the number of instances (pages) in our training data. We employ a maximum likelihood estimation technique to estimate parameters that maximize the log-likelihood.

Finally, decoding is the task of finding the optimal label sequence y^* that achieves the highest conditional probability:

$$y^* = \underset{y \in \mathscr{Y}(h)}{\operatorname{argmax}}\, p(y|h;\theta) \tag{10}$$

We find the most probable label sequence by using the Viterbi algorithm. Here the maximization occurs over all of the possible labels for each stroke.

Table II: Model configuration

Layer no.	Type	Specifications
1	Convolution	Filters = 16, Kernel size = 9, strides = 1
2	Convolution	Filters = 32, Kernel size = 9, strides = 1
3	BLSTM	Output dimension = 64
4	BLSTM	Output dimension = 64
5	BLSTM	Output dimension = 64

Table III: Frequencies of stroke types in each dataset.

Dataset	Stroke Type		
	FBD	Cross-out	Text
Training Data	31.2%	1.2%	67.6%
Testing Data	20.5%	1.4%	78.1%

IV. System workflow

We preprocess the pen strokes before feeding them to the neural network. For the input sequence I_1, we use min-max normalization to linearly transform the x and y coordinate values to map into the range $[0,1]$. Likewise, we use z-normalization to normalize the feature values in the input sequence I_2.

As shown in Figure 2, I_1 and I_2 are fed to the right and left CNN-BLSTM networks, respectively. We observed that the BLSTM networks easily overfit the training data. As a remedy, we apply dropout to the recurrent input signal on the BLSTM networks. During network training, the dropout rate is set to 0.2. The configuration of the CNN-BLSTM networks is shown in Table II. Both of the CNN-BLSTM networks utilize the same hyper parameters. The output vectors of the BLSTM networks are combined together and is used as the input to the CRF layer. Finally, the CRF layer jointly predicts the best label sequence for the output nodes.

A. Network Training

We use Keras [15] to implement our neural network model. The entire model contains 216,996 trainable parameters. The training and testing experiments were run on a machine equipped with 2.66 GHz Xeon(R) CPU. For the configuration described above, the model training requires approximately 49 hours.

B. Optimization method

We use RMSProp [16] to optimize the parameters of the network. At each step of network training we feed batches of size 10 to the network. The learning rate is initially set to 0.001 and at each step is updated according to [16]. Our model achieves the highest accuracy on the validation set (we discuss the validation set in Section V) at 278 epochs.

V. Experiment

We evaluate the performance of our model on a database of homework assignments, quizzes, and exams collected from 132 undergraduate students enrolled in a mechanical engineering course on statics. The students produced the writing using LiveScribe digital pens. These pens are used with special dot-patterned paper. A camera integrated into the tip of the pen uses the dots to digitize the writing as time-stamped coordinates.

Table IV: Performance of our model and three base line networks.

Model	Accuracy	Precision	Recall	F1 Score
CNN-BLSTM$_1$ $-$ *CRF*	88.30%	88.42%	88.30%	88.36%
CNN-BLSTM$_2$ $-$ *CRF*	91.34%	91.43%	91.22%	91.32%
CNN-BLSTM	90.76%	89.51%	90.46%	89.98%
CNN-BLSTM-CRF	**94.70%**	**96.14%**	**94.54%**	**95.33%**

6,562 pages of handwritten coursework were collected from 12 exam problems, 30 homework problems, and 7 quiz problems. From this, we manually labeled 1,060 pages comprising solutions to 5 exam problems (293 pages) and 8 homework problems (776 pages). The exam pages contained 122,058 pen strokes and homework pages contained 298,527 pen strokes. The frequencies of the various stroke types labeled exam and homework pages are shown in Table III. We used the homework data for training and the exam data for testing. We held out 155 pages from the training set for validation.

A. Results for Various System Configurations

To evaluate the power of each part of our model, we computed the accuracy with various parts of the model removed. More specifically, we considered 3 sub-models: (1) The CNN-BLSTM$_1$-CRF model uses only the I_1 input sequence (i.e., the pen stroke coordinates) for input and excludes the CNN-BLSTM network on the right of the network (i.e., the domain-dependent features). (2) Conversely, the CNN-BLSTM$_2$-CRF model uses only the I_2 input sequence and ignores the CNN-BLSTM network on the left side. (3) The CNN-BLSTM model removes the CRF layer and replaces it with a dense layer. All of these networks utilize the same hyper-parameters as displayed in Table II. As shown in Table IV, our complete model achieved higher accuracy than any of the sub-models, indicating the importance of all of the elements of our model.

B. Comparison with Related Methods

We benchmark our model with three existing methods: (1) The method of Stahovich and Lin ; (2) the GSC26_BCC26_19Q method [14]; and (3) the CRF_NN [8]. The method first of these methods was specifically designed for free-form statics solutions, while the other two are the top-performing systems for classifying strokes into text/non-text classes on the IAMonDo database [17].

Table V compares the performance of our method to that of the three benchmark methods on the testing dataset. Our method performed better than the GSC26_BCC26_19Q

Table V: Performance of models for stroke classification task.

Model	Accuracy	Precision			Recall		
		FBD	Cross-out	Text	FBD	Cross-out	Text
Our Model	**94.7%**	**88.57%**	15%	96.14%	**87.88%**	0.68%	**97.00%**
Stahovich and Lin [1]	92.23%	77.32%	**53.91%**	**97.26%**	87.74%	**74.26%**	93.59%
CRF_NN [8]	89.55%	71.84%	9.49%	93.95%	78.24%	0.97%	93.66%
GSC26_BCC26_19Q [14]	91.33%	79.97%	27.65%	93.61%	74.81%	0.74%	96.61%

and CRF_NN methods on all performance measures. Our method also achieved the highest overall accuracy of 94.7%. For the most part, our method performed better than the method from [1] for FBD and text strokes: Our method achieved higher recall rates for both types of strokes and achieved a higher precision rate for FBD strokes. The precision of our method for text strokes was only slightly less (about one percentage point) than that of the method from [1]. Due to the small number of cross-out instances in the training set, most of the systems have low performance in recognizing cross-out strokes. However, [1] uses a special purpose, hand-coded cross-out detection technique that achieves high accuracy for this class.

VI. Conclusion

In this paper we present a novel model for classifying strokes of online handwritten documents into FBD, cross-out, and text classes. The model is composed of two channels for extracting features and encoding information from pen trajectories and stroke features. The outputs of these channels are concatenated together to form the input to a CRF layer that predicts the best classes for the strokes. We show that our method outperforms other state-of-the-art methods for the stroke classification task.

Acknowledgement

Shelton was supported by the National Science Foundation (IIS 1510741).

References

[1] T. F. Stahovich and H. Lin, "Enabling data mining of handwritten coursework," *Computers & Graphics*, vol. 57, pp. 31–45, 2016.

[2] K. Jain, A. M. Namboodiri, and J. Subrahmonia, "Structure in on-line documents," in *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*. IEEE, 2001, pp. 844–848.

[3] A. Delaye and K. Lee, "A flexible framework for online document segmentation by pairwise stroke distance learning," *Pattern Recognition*, vol. 48, no. 4, pp. 1197–1210, 2015.

[4] T. F. Stahovich, E. J. Peterson, and H. Lin, "An efficient, classification-based approach for grouping pen strokes into objects," *Computers & Graphics*, vol. 42, pp. 14–30, 2014.

[5] C. M. Bishop, M. Svensen, and G. E. Hinton, "Distinguishing text from graphics in on-line handwritten ink," in *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*. IEEE, 2004, pp. 142–147.

[6] X.-D. Zhou and C.-L. Liu, "Text/non-text ink stroke classification in japanese handwriting based on markov random fields," in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, vol. 1. IEEE, 2007, pp. 377–381.

[7] A. Delaye and C.-L. Liu, "Context modeling for text/non-text separation in free-form online handwritten documents," in *Document Recognition and Retrieval XX*, vol. 8658. International Society for Optics and Photonics, 2013, p. 86580C.

[8] J.-Y. Ye, Y.-M. Zhang, and C.-L. Liu, "Joint training of conditional random fields and neural networks for stroke classification in online handwritten documents," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 3264–3269.

[9] M. Shilman, Z. Wei, S. Raghupathy, P. Simard, and D. Jones, "Discerning structure from freeform handwritten notes," in *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*. IEEE, 2003, pp. 60–65.

[10] J. Blanchard and T. Artieres, "On-line handwritten documents segmentation," in *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*. IEEE, 2004, pp. 148–153.

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

[12] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," *arXiv preprint arXiv:1505.08075*, 2015.

[13] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[14] T. Van Phan and M. Nakagawa, "Combination of global and local contexts for text/non-text classification in heterogeneous online handwritten documents," *Pattern Recognition*, vol. 51, pp. 112–124, 2016.

[15] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[16] Y. Dauphin, H. de Vries, and Y. Bengio, "Equilibrated adaptive learning rates for non-convex optimization," in *Advances in neural information processing systems*, 2015, pp. 1504–1512.

[17] E. Indermühle, M. Liwicki, and H. Bunke, "Iamondo-database: an online handwritten document database with non-uniform contents," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. ACM, 2010, pp. 97–104.