Transcribing Content from Structural Images with Spotlight Mechanism

Yu Yin, Zhenya Huang
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China
{yxonic,huangzhy}@mail.ustc.edu.cn

Enhong Chen*
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China
cheneh@ustc.edu.cn

Qi Liu
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China
qiliuql@ustc.edu.cn

Fuzheng Zhang, Xing Xie
Microsoft Research Asia
{fuzzhang,xing.xie}@microsoft.com

Guoping Hu
iFLYTEK Research
gphu@iflytek.com

ABSTRACT

Transcribing content from structural images, e.g., writing notes from music scores, is a challenging task as not only the content objects should be recognized, but the internal structure should also be preserved. Existing image recognition methods mainly work on images with simple content (e.g., text lines with characters), but are not capable to identify ones with more complex content (e.g., structured code), which often follow a fine-grained grammar. To this end, in this paper, we propose a hierarchical *S*potlight *T*ranscribing *N*etwork (STN) framework followed by a two-stage "where-to-what" solution. Specifically, we first decide "where-to-look" through a novel spotlight mechanism to focus on different areas of the original image following its structure. Then, we decide "what-to-write" by developing a GRU based network with the spotlight areas for transcribing the content accordingly. Moreover, we propose two implementations on the basis of STN, i.e., STNM and STNR, where the spotlight movement follows the Markov property and Recurrent modeling, respectively. We also design a reinforcement method to refine our STN framework by self-improving the spotlight mechanism. We conduct extensive experiments on many structural image datasets, where the results clearly demonstrate the effectiveness of STN framework.

KEYWORDS

Structural image; Spotlight Transcribing Network; reinforcement learning

ACM Reference Format:
Yu Yin, Zhenya Huang, Enhong Chen, Qi Liu, Fuzheng Zhang, Xing Xie, and Guoping Hu. 2018. Transcribing Content from Structural Images with Spotlight Mechanism. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3219962

*The corresponding author.

KDD '18, August 19–23, 2018, London, United Kingdom

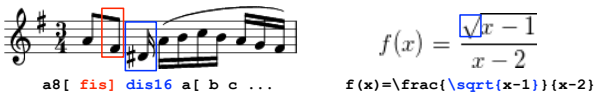
ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3219962

1 INTRODUCTION

Transcribing content from images refers to recognizing semantic information in images into comprehensible forms (e.g., text) in computer vision [38]. It is an essential problem for computers to understand how humans communicate about what they see, which includes many tasks, such as reading text from scenes [17, 40], writing notes from music scores [28] and recognizing formulas from pictures [6]. As it is crucial in many applications, e.g., image retrieval [5, 29], online education systems [13, 20] and assistant devices [9], much attention has been attracted from both academia and industry [38].

In the literature, there are many efforts for this transcribing problem, especially on text reading task. Among them, the most representative one called Optical Character Recognition (OCR) has been extensively studied in many decades [14], which mainly follows rule-based solutions for generating texts from well-scanned documents [21]. Recently, researchers focus on a more general scene text recognition task, aiming to recognize texts from natural images [33]. Usually, existing approaches are designed in an encoder-decoder architecture, which consists of two components: (1) a CNN based encoder to capture and represent images as feature vectors that preserve their the semantic information [26]; (2) a RNN based decoder that decodes the features and generates output text sequences either directly [33], or attentively [36]. Though good performances have been achieved, previous studies mainly focus on the images with straightforward content (i.e., text with characters), while ignoring large proportion of structural images, where the content objects are well-formed in complex manners, e.g., music scores (Figure 1(a)) and formulas (Figure 1(b)). Therefore, the problem of transcribing content from these structural images remains pretty much open.

In fact, there are many technical challenges along this line due to the unique characteristics of structural images. First, different from natural images, where the text content is mostly placed in simple patterns, in structural images, the content objects usually follow a fine-grained grammar, and are organized in a more complex manner. E.g., in Figure 1(a), notes from the music score are not only



(a) Music score example (b) Formula example

Figure 1: Some structural image examples. Left is a music bar from Cello Suite No. 1 in G major by Bach; Right is a function formula from a high school math exercise.

placed simply from left to right, but the positions in the stave for each note are also specified, often with annotations added left or above. A division formula in Figure 1(b) contains nested structure, where the equation components are placed at the left and right side of the equal sign, with two parts of the right-hand-side fraction placed above and below the middle line. Thus, it is necessary for transcribing to not only capture the information from local areas, but also preserve the internal structure and organization of the content. Second, content objects in structural images, even if they just take a small proportion, may carry much semantics. For example, the note marked by blue box in Figure 1(a) is written as "`dis16`" in LilyPond[1], which means that the note is D# ("`-is`" for sharp), and the note is a sixteenth note (denoted by "`16`"); the formula marked in Figure 1(b) means "`\sqrt{...}`" in TEX code, representing the square root operator, with the scope defined by curly braces. Thus, it is very challenging to transcribe the complete content from an area containing such a informative object, compared to just one character in tasks such as scene text recognition. Third, there exist plenty of similar objects puzzling the transcribing task, e.g., a sixteenth note (blue in Figure 1(a)) just contains one more flag on the stem than an eighth note (red), while notes with same duration and different pitches are almost identical except for their positioning. This characteristic requires a careful design for the transcribing.

To address the above challenges, following the observation on human transcribing process, i.e., first find out where to look, then write down the content, we present a two-stage "where-to-what" solution and propose a hierarchical framework called the *S*potlighted *T*ranscribing *N*etwork (STN) for transcribing content from structural images. Specifically, after encoding images as features vectors, in our decoder component, we first propose a spotlight module with a novel mechanism to handle the "where-to-look" problem and decide a reading path focusing on areas of the original image following its internal structure. Then, based on the learned spotlights areas, we aim for "what-to-write" problem and develop a GRU based network for transcribing the semantic content from the local spotlight areas. Moreover, we propose two implementations on the basis of the STN framework. The first is a straightforward one, i.e., *STNM with Markov property*, in which the spotlight placement follows a Markov chain. Comparatively, the second is a more sophisticated one, i.e., *STNR with Recurrent modeling*, which can track long-term characteristics of spotlight movements. We also design a reinforcement method to refine STN, self-improving the spotlight mechanism. We conduct extensive experiments on real-world structural image datasets, where the results clearly demonstrate the effectiveness of the STN framework.

[1]A domain specific language for music notation, http://lilypond.org/

2 RELATED WORK

The related research topics to our concerns can be classified into the following three categories: encoder-decoder system, attention mechanism, and reinforcement learning.

2.1 Encoder-Decoder System

The encoder-decoder system is a general framework, which has been applied to many applications, such as neural machine translation [3, 7] and image captioning [33, 36]. Generally, the system has two separate parts, one encoder for representing and encoding the input information into a feature vector, and one decoder for generating the output sequence according to the encoded representation. Due to its remarkable performance, many efforts have been made to apply it to scene text recognition [35], aiming at transcribing texts from natural images. Specifically, for encoder design, representative works leveraged deep CNN based networks, which have been the most popular methods due to their performance on hierarchical feature extraction [26], to learn the information encodings from images [16]. Then for decoder selection, variations of recurrent neural networks (RNN), such as LSTM [12] and GRU [8], were utilized to generate the output text sequence, both of which are able to preserve long-term dependencies for text representations [32]. The whole architecture is end-to-end, which show the effectiveness in practice [30].

2.2 Attention Mechanism

However, in the original encoder-decoder systems, encoding the whole input into one vector usually makes the encoded information of images clumsy and confusing for the decoder to read from, leading to unsatisfactory transcription [22]. To improve the encoder-decoder models addressing this problem, inspired by human visual system, researchers have tried to propose many attention mechanisms to highlight different parts of the encoder output by assigning weights to encoding vectors in each step of text generation [3, 24, 36] or sequential prediction [31, 39]. For example, Bahdanau et al. [3] proposed a way to jointly generate and align words using attention mechanism. Xu et al. [36] proposed soft and hard attention mechanisms for image captioning. Lee et al. [19] used an attention-based encoder-decoder system for character recognition problems.

Our work improves the previous studies mainly from the following two aspects. First, the attention weights are usually calculated by the correspondence between outputs and the whole content, which let the models know "what" to look but not "where" to look. In our work, we propose a novel spotlight mechanism to directly find a reading path tracking the image structure for transcribing. Second, previous decoding process has one RNN for learning attentions and transcribing simultaneously, which may cause some confusion for transcription, while our framework models spotlighting and transcribing with two separate facilities, avoiding the confusion between two sequences.

2.3 Reinforcement Learning

Deep reinforcement learning is a kind of state-of-the-art technique, which has shown superior abilities in many fields, such as gaming and robotics [1]. The main idea of them is to learn and refine model

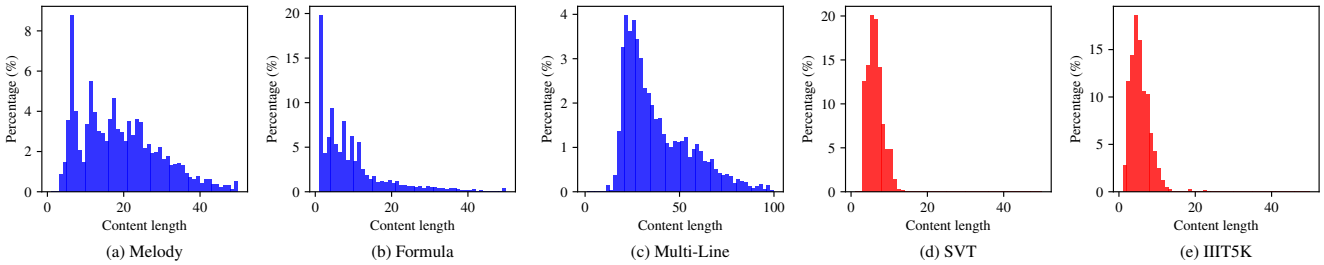


Figure 2: Comparison of structural image (blue) and scene text recognition datasets (red) on content length distribution.

Table 1: The statistics of the datasets.

Dataset	Image count	Token space	Token count	Avg. tokens per image	Avg. image pixels
Melody	4208	70	82,834	19.7	15,602.7
Formula	61649	127	607,061	9.7	1,190.7
Multi-Line	4595	127	182,112	39.8	9,016.6
SVT	618	26	3,796	5.9	12,733.5
IIIT5K	3000	36	15,269	5.0	11,682.0

parameters according to task-specific reward signals. For example, Ranzato et al. [27] used the whole sequence metrics to guide the sequence generation, using REINFORCE method; Bahdanau et al. [2] utilized the actor-critic algorithm for sequence prediction, refining the model to improve sentence BLEU score.

3 PRELIMINARIES

In this section, we first give a clear definition of structural images, and introduce the structural image datasets used in this paper. Then we discuss the crucial differences between structural image transcribing and typical scene text recognition with exclusive data analysis. At last, we give the formal definition of the structural image transcription problem.

3.1 Data Description

In this paper, we mainly focus on transcribing content from structural images. *Structural images* refer to printed graphics that are not only a set of content objects, but also contain meaningful structure, i.e., object placement, following a certain grammar. Content with its structure can often be described by a domain specific language and complied by the corresponding software. Typical structural images include music scores, formulas and flow charts, etc., which can be described in music notation, TEX and UML code, respectively.

We exploit two real-world datasets, i.e., *Melody* and *Formula*, along with one synthetic dataset *Multi-Line*, specifically for the structural image transcription task[2]. The *Melody* dataset contains pieces of music scores and their source code in LilyPond collected from the Internet[3], mostly instrumental solos and choral pieces written by Bach, split into 1 to 4 bar length, forming 4208 image-code pairs. The *Formula* dataset is collected from Zhixue.com, an online educational system, which contains 61649 printed formulas from high school math exercises, with their corresponding TEX code. To further demonstrate transcription on images with more complicated structure, we also construct the *Multi-Line* dataset that contains 4595 multi-line formulas, e.g., piecewise function, each line consisting of some complex formulas, e.g., multiple integral. We summarize some basic statistics of these datasets in Table 1.

We now conduct deep analysis to show the unique characteristics of the structural image transcription task compared to traditional scene text recognition. Specifically, we compare our datasets with two commonly used datasets for scene text recognition, i.e., SVT [34] and IIIT5K [23], and conclude three main differences. First, structural image transcription needs to preserve more information: other than just objects, how they are organized should also be transcribed. As shown in Table 1 and Figure 2, our datasets contain significantly longer content in relatively small images. Sequences longer than 10 tokens taking 75.0%, 30.4% and 99.9% of Melody, Formula and Multi-Line datasets, respectively. However, only 1.9% in SVT and 2.7% in IIIT5K have more than 10 character long sequences. In addition, Melody, Formula and Multi-Line contain in average 1.26, 8.15 and 4.14 tokens every 1000 pixels, while SVT and IIIT5k only contain 0.46 and 0.43 characters, respectively, which indicates that each proportion of an image contains more information to be transcribed, along with the informative structure. Second, the output space and count in our datasets are often larger than SVT and IIIT5K, as shown in Table 1. Hence, it is even more complicated to transcribe content from structural images compared to text recognition. Third, structural image transcription process is reversible, meaning the corresponding code should be able to compile and regenerate the original image, which is not necessary or possible for traditional scene text recognition.

In summary, the above analysis clearly shows that the structural image transcription problem is quite different from traditional scene text recognition tasks. As a result, it is necessary to design a new approach that better fits this problem.

3.2 Problem Definition

In this subsection, we formally introduce the structural image transcription problem. In our image transcribing applications, we are given structural images and their corresponding source code. Each

[2]Datasets are available at: http://home.ustc.edu.cn/~yxonic/stn_dataset.7z.
[3]http://web.mit.edu/music21/

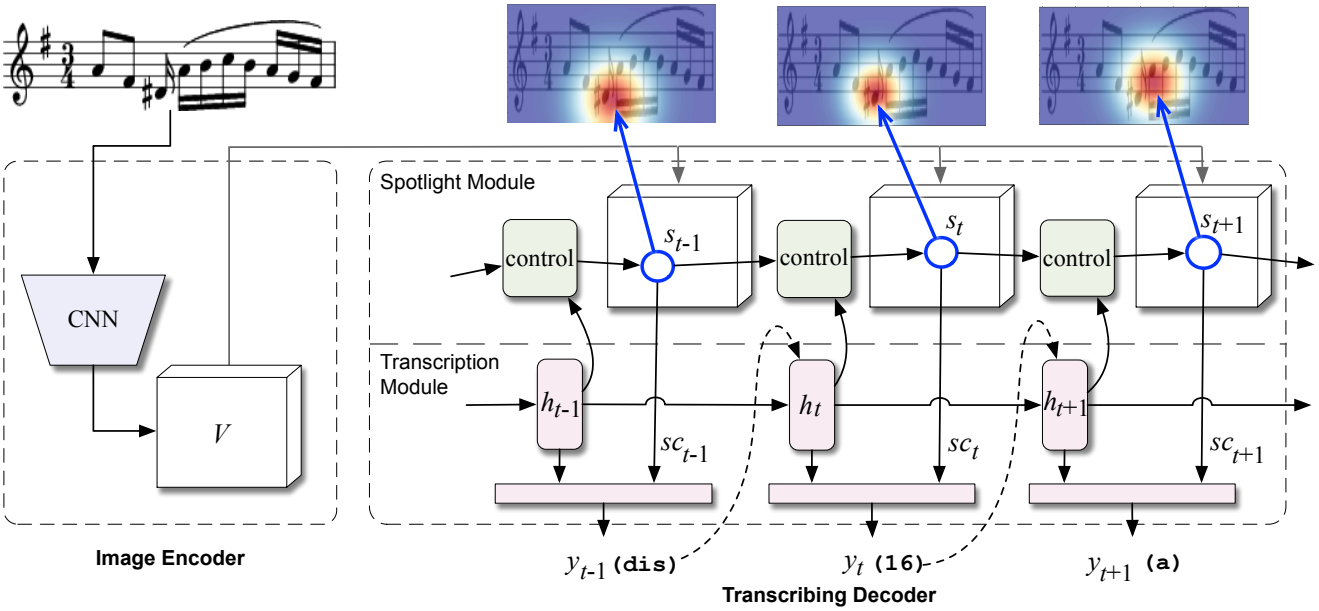


Figure 3: The STN model architecture consists of two main parts: 1) a convolutional image feature extractor as the encoder, and 2) the transcribing decoder. At the decoding stage, the spotlight module is first engaged to handle the "where-to-look" problem. Afterwards, the transcription module finds out "what-to-write" by utilizing the spotlighted information from the encoder, generating the transcribed content one token at a time.

input image x is a one-channel gray-scale image with width W and height H, containing content such as music notations or printed formulas. For each image, the expected output, i.e., its source code, is given as a token sequence $y = \{y_1, y_2, \ldots, y_T\}$, where T is the length of token sequence. Each y_t can be a LilyPond notation (`c`, `fis`, ...) in music score transcribing task, or a TEX token (`x`, `\frac`, ...) in formula transcribing task. Moreover, structural images are reversible, by which we mean that the token sequence is expected to reconstruct the original image using the corresponding compiler. Therefore, the problem can be defined as:

Definition 3.1. (**Structural Image Transcription Problem**). Given a structural $W \times H$ image x, our goal is to transcribe the content from it as a sequence $\hat{y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_T\}$ as close as possible to the source code sequence y, where each $\hat{y}_t$ is the predicted token taking from the specific language corresponding to the image.

4 SPOTLIGHTED TRANSCRIBING NETWORK

In this section, we introduce the Spotlighted Transcribing Network (STN) framework in detail. First we give an overview of the model architecture. Then we describe all the details of our proposed spotlight mechanism in following sections. Finally we discuss the training process of STN with reinforcement learning for refinement.

4.1 Model Overview

Figure 3 shows the overall architecture of Spotlighted Transcribing Network (STN), which consists of two main components: (1) a convolutional feature extractor network as the encoder, which learns the visual representations V from the input image x; (2) a hierarchical transcribing decoder, which we mainly focus on in this work. Mimicking human reading process, the decoder first takes the encoded image information V and find out "where-to-look" by shedding spotlight on it, following the learned reading path, then generates the token sequence y, by predicting one token at a time using a GRU-based output network, solving the "what-to-write" problem. In the following subsections, we will explain how each part of the STN works in detail.

4.2 Image Encoder

The encoder part of STN is for extracting and embedding information from the image. Instead of embedding the complete image x into one vector, which may cause a loss in structural information [36], we extract a set of feature vectors V, each of which is a D-dimensional representation corresponding to a part of the image:

$$V = \{V^{(i,j)} : i = 1, \ldots, W', j = 1, \ldots, H'\}, V^{(i,j)} \in \mathbb{R}^D.$$

A deep convolutional neural network (CNN) is used as the feature extractor to capture high-level semantic information, which we denote as $f(\cdot; \theta_f)$. We follow the state-of-the-art image feature extractor design as in ResNet [11], adding residual connections between convolutional layers, together with ReLU activation [25] and batch normalization [15] to stabilize training, but removing the fully connected layers along with higher convolutional and pooling layers. As a result, we construct an extractor network that takes an image x, outputs a 3 dimensional tensor V ($W' \times H' \times D$):

$$V = f(x; \theta_f), \quad (1)$$

where vector $V^{(i,j)}$ at each location (i, j) represents the local semantic information. The output tensor also preserves spatial and contextual information, with the property that adjacent vectors representing neighboring parts of the image. This allows the decoder module to use the image information selectively with both content and location in mind.

4.3 Transcribing Decoder

The transcribing decoder of STN, as in typical encoder-decoder architecture, generates one token at a time, by giving its conditional probability over the encoder output V and all the previous outputs $\{y_1, \ldots, y_{t-1}\}$ at each time step t. Hence, we can denote the probability of a decoder yielding a sequence y as:

$$\mathrm{P}(y|x) = \prod_{t=1}^{T} \mathrm{P}(y_t|y_1, \ldots, y_{t-1}, V). \tag{2}$$

Considering the fact that the output history can be long, we embed the history before time step t into a hidden state vector h_t by utilizing a variation of RNN — Gated Recurrent Unit (GRU), which preserves more long-term dependencies. Formally, at time step t, the hidden state for output history h_t is updated based on the last output item y_{t-1} and the previous output history h_{t-1}, by an GRU network $GRU(\cdot\,; \theta_h)$:

$$h_t = GRU(y_{t-1}, h_{t-1}; \theta_h). \tag{3}$$

For image part, the visual representation V we get as the encoder output carries enough semantic information, but as a whole it can be confounding for the decoder to comprehend, and thus needs careful selection [36]. To deal with this problem, we mimic what human do when reading images: focus on one spot at a time, write down content, then focus on a next spot following the image structure [4]. Along this line, we propose a module with novel spotlight mechanism, where at each time step, we only focus on information around a certain spotlight center. We refer to the spotlight center position as s_t at time step t, and the spotlighted information as spotlight context sc_t. Further details on how to get focused spotlight context are described in Section 4.4, while how to move the spotlight following the structure is described in Section 4.5.

With embedded history h_t, and spotlight context sc_t, together with current spotlight position s_t, the conditional probability of output token at time t can then be parameterized as follows:

$$\mathrm{P}(y_t|y_1, \ldots, y_{t-1}, V) = \mathrm{Softmax}(d(h_t \oplus sc_t \oplus s_t; \theta_d)), \tag{4}$$

where $d(\cdot\,; \theta_d)$ is a transformation function (e.g. a feed-forward neural network) that outputs a vocabulary-sized vector, and $\oplus$ represents the operation that concatenates two vectors. The overall transcription loss $\mathcal{L}$ on an image-sequence pair is then defined as the negative log likelihood of the token sequence over the image:

$$\mathcal{L} = \sum_{t=1}^{T} -\log P(y_t|y_1, \ldots, y_{t-1}, V). \tag{5}$$

With all the calculation being deterministic and differentiable, the model can be optimized through standard back-propagation.

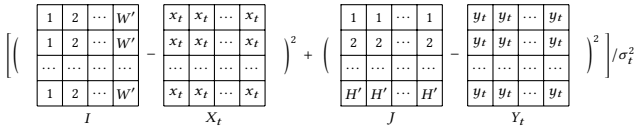


Figure 4: Demonstration of the parallelized operation on assigning weights. It should be clear that the element at each position (i, j) of the result matrix is $[(i - x_t)^2 + (j - y_t)^2]/\sigma_t^2$.

4.4 Spotlight Mechanism

In this subsection, we describe how to get focused information of the input image, i.e., the spotlight context sc_t, with our proposed spotlight mechanism. How the spotlight moves through time is handled in a separate spotlight control module, and is described later in detail in Section 4.5.

As mentioned earlier, the visual embedding V is confounding for the decoder, and we want to focus on one spot at a time when generating output. To achieve this goal, we propose a novel spotlight mechanism to mimic human focus directly, where at each time step, we only care about information around a certain location which we call a spotlight center, by "shedding" a spotlight around it. More specifically, we define a spotlight handle $s_t = (x_t, y_t, \sigma_t)^{\mathrm{T}}$ at each time step t to represent the spotlight, where (x_t, y_t) represents the center position of the spotlight, and σ_t represents the radius of the spotlight. Inspired by Yang et al. [37], we "shed" a spotlight by assigning weights to image representation vectors at each position, following a truncated Gaussian distribution centered at (x_t, y_t), with the same variance σ_t on both axis.

Formally, under the spotlight with handle $s_t = (x_t, y_t, \sigma_t)^{\mathrm{T}}$, the weights for each vector at position (i, j) at time step t, denoted as $\alpha_t^{(i,j)}$, is proportional to the probability density at point (i, j) under Gaussian distribution:

$$\alpha_t^{(i,j)} \sim \mathcal{N}((i, j)^{\mathrm{T}}|\mu_t, \Sigma_t), \tag{6}$$

$$\mu_t = (x_t, y_t)^T \quad \Sigma_t = \begin{bmatrix} \sigma_t & 0 \\ 0 & \sigma_t \end{bmatrix}. \tag{7}$$

Intuitively, the closer (i, j) is to the center (x_t, y_t), the higher the weight should be, mimicking shedding a spotlight with radius σ_t onto the location (x_t, y_t). To calculate the weight $\alpha_t^{(i,j)}$ of each position (i, j) while still make the process differentiable, we apply the definition of Gaussian distribution and rewrite the expression of $\alpha_t^{(i,j)}$ as:

$$\alpha_t^{(i,j)} = \mathrm{Softmax}(b_t) = \frac{\exp(b_t^{(i,j)})}{\sum_{u=1}^{W'} \sum_{v=1}^{H'} \exp(b_t^{(u,v)})}, \tag{8}$$

$$b_t^{(i,j)} = -\frac{(i - x_t)^2 + (j - y_t)^2}{\sigma_t^2}, \tag{9}$$

where b measures how close the point (i, j) is to the center (x_t, y_t), i.e., how important this point is, and α is thus a $W' \times H'$ matrix following the truncated Gaussian distribution for each point (i, j), and can later be used as weights for each image feature vector.

To parallize the calculation of Equation (9), we perform a small trick as demonstrated in Figure 4. We first construct two $W' \times H'$ matrices I and J in advance, each of them representing one

coordinate. Specifically, as shown in Figure 4, for each point (i, j), we have $I^{(i,j)} = i$ and $J^{(i,j)} = j$. We also expand x_t and y_t as $W' \times H'$ matrices X_t and Y_t respectively, with same value for each element. Therefore, Equation (9) can be written as the matrix form:

$$b_t = -[(I - X_t)^2 + (J - Y_t)^2]/\sigma_t^2 \tag{10}$$

The focused information of the visual representation V at time step t can then be computed as a spotlight context vector sc_t weighted by $\alpha_t^{(i,j)}$ according to current spotlight handle s_t, i.e., the weighted sum of features at each position:

$$sc_t = \sum_{i=1}^{W'} \sum_{j=1}^{H'} \alpha_t^{(i,j)} V^{(i,j)} \tag{11}$$

Please note that the spotlight context sc_t represents the information in the focused area at time step t, and should contain useful information specifically for transcribing at current time step. By focusing directly on the correct spot, the transcription module therefore only cares about the local information, not confusing at areas with similar content all over the image.

4.5 Spotlight Control

Now we discuss how to control the spotlight to find a proper reading path, following the image structure through the whole generation process. Different from traditional attention strategy where both output sequence and attention behavior are embedded in one module, we see the spotlight movement (i.e., the value of the spotlight handle $s_t = (x_t, y_t, \sigma_t)^{\mathrm{T}}$ at each time step t) as a separate sequence devoted to following the image structure, and model this sequence with a standalone spotlight controlling module, without mixing the information with the output sequence. We provide two implementations under the STN framework, i.e., the straightforward *STNM with Markov property*, and the more sophisticated *STNR with Recurrent modeling*, utilizing another GRU network. Each implementation models the spotlight handle sequences differently.

STNM with Markov property. With an assumption that is not far from reality, we can intuitively treat the spotlight handle sequence as a Markov process, i.e., current spotlight handle only depends on the previous handle, along with other internal states at current time step. Treating the spotlight handle as a Markov process means the probability of choosing s_t at time t does not rely on spotlight handles more than one step earlier, i.e.:

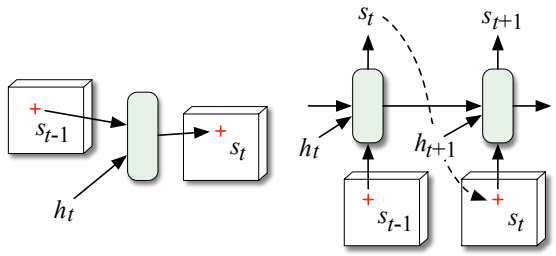
$$P(s_t|s_1, \ldots, s_{t-1}; \cdot) = P(s_t|s_{t-1}; \cdot). \tag{12}$$

To decide where to put the spotlight properly, the model also needs to know current internal states at time step t, including the spotlight context sc_{t-1} which represents previous spotlighted region, and the history embedding h_t which represents output history *before* time t. Thus, we can use a feed-forward neural network $n(\cdot\,; \theta_n)$ to model the choice of s_t (Figure 5 (a)) as:

$$s_t = n(s_{t-1} \oplus sc_{t-1} \oplus h_t; \theta_n) \tag{13}$$

The way we model the sequence is simple and time-independent, which makes it easier for the controlling module to train.

STNR with Recurrent modeling. Sometimes longer spotlight history is needed for spotlight controlling on images with more complex structure. To track the image structure as a sequence with long-term dependency, we propose another GRU network



(a) Markovian control module. (b) Recurrent control module.

Figure 5: The spotlight control module implementations.

$GRU(\cdot\,; \theta_g)$ to track the spotlight history, and a fully connected layer $c(\cdot\,; \theta_c)$ to generate next spotlight handle (Figure 5 (b)). Specifically, at time step t, with last spotlight history embedding denoted as e_t, the current spotlight handle s_t at time t is calculated as:

$$s_t = c(e_t \oplus sc_{t-1} \oplus h_t; \theta_c) \tag{14}$$

and the history embedding is updated by:

$$e_t = GRU(s_{t-1}, e_{t-1}; \theta_g) \tag{15}$$

Through a separate module specifically for spotlight control, STN gains two advantages over the traditional attention mechanism. First, STN focuses on local areas by design, and the model will only have to learn where to focus and what to transcribe, while the attention model have to first learn to focus, then learn what to focus on. Second, modeling reading and writing process as two separate sequences, with a standalone module dedicated for the "where-to-look" problem, STN is capable for directly learning a reading path on structural images apart from generating the output sequences, which enables our model to track the image structure more closely compared to attentive models where attentions and transcribing process are modeled together in only one network.

4.6 Training and Refining STN

Parameters to be updated in both implementations comes from three parts: the encoder parameters θ_f, the decoder parameters $\{\theta_h, \theta_d\}$, and parameters in the spotlight control module, which are θ_n in STNM and $\{\theta_c, \theta_g\}$ in STNR. The parameters are updated to minimize the total transcription loss $\mathcal{L}$ (Equation (5)) through a gradient descent algorithm, which we choose the Adam optimizer [18]. More detailed settings are presented in the experiment section.

Though our model is differentiable, and can be optimized through back-propagation methods, directly training to fit the label suffers from some specific aspects in the image transcribing task. Firstly, the model has to jointly learn two different sequences with only one of them directly supervised, which may result in inaccurate reading path. Second, the given token sequence may only be one of the many correct ones that all regenerates the original image. For instance, in LilyPond notation, we can optionally omit duration for notes at same length with their predecessors. Fitting to only one of the correct sequences lets down the model even when it achieves good strategies. Fortunately, in structural image transcription problems, we have an advantage that the process is reversible, meaning given the transcribed sequence, we can use a compiler to reconstruct

the image. With the guidance of this, we can further refine our model using reinforcement learning, by regarding our sequential generation as a decision making problem, viewing it as a Markov Decision Process (MDP) [2]. Formally, we define the *state*, *action* and *reward* of the MDP as follows:

State: View our problem as outputting the probability of items at each time step conditioned by the image and previous generations, the environment state at time step t as the combination of the image x and the output history $\{y_1, \ldots, y_{t-1}\}$, which is exactly the inputs of the STN. Therefore, instead of directly using the environment state, we use the internal states (combined and denoted as $state_t$) in STN framework as MDP states.

Action: Taking action a_t is defined as generating the token y_t at time step t. With the probability of each token as the output, the STN can be viewed as a stochastic policy that generates actions by sampling from the distribution $\pi(a|state_t; \theta) = P(a|y_1, \ldots, y_{t-1}, x; \theta)$, where θ is the set of model parameters to be refined.

Reward: After taking the action, a reward signal r is received. Here we define the reward r_t as 0 when the generation is not finished at time step t, or the pixel similarity between the reconstruction image and the original image after the whole generation process finished. Besides, we give -1 as the final reward if the output sequence does not compile, addressing grammar constraints by penalizing illegal outputs. The goal is to maximize the sum of the discounted rewards from each time t, i.e., the return:

$$R_t = \sum_{k=t}^{T} \gamma^k r_k. \tag{16}$$

We further define a value network $v(\cdot\,; \theta_v)$ for estimation of the expected return from each $state_t$, which is a feed-forward network with the same input as the STN output layer d. The estimated value v_t, i.e., the expected return, at time step t is then

$$v_t = v(h_t \oplus sc_t \oplus s_t; \theta_v). \tag{17}$$

With a stochastic policy together with a value network, we can apply the actor-critic algorithm [2] to our sequence generation problem, with the policy network trained using policy gradient at each time step t as:

$$\nabla_\theta = \log \pi(a|state_t; \theta)(R_t - v_t), \tag{18}$$

and the value network trained by optimizing the distance between the estimated value and actual return: $\mathcal{L}_{value} = ||v_t - R_t||_2^2$.

As the whole model is complicated, directly applying reinforcement learning to the model suffers from the large searching space. Through experiments we notice that, after supervised training, the image extractor and the output history embedding modules have both been trained properly, and it is more important for our framework to have a better reading path to make precise predictions, which indicates that refining the spotlight module is most beneficial. Therefore, at reinforcement stage, we only optimize parameters from the spotlight control module (θ_n in STNM, θ_c and θ_g in STNR), along with those from the output layer (θ_o), and omit θ_f and θ_h, which reduces the variance when applying reinforcement learning algorithms, and get better improvements.

With this train-and-refine procedure, our model can learn a reasonable reading path on structural images, focusing on different parts following the image structure when transcribing, and get superior transcription results, as our experimental results show in the next section.

5 EXPERIMENTS

In this section, we conduct extensive experiments to demonstrate the effectiveness of STN model from various aspects: (1) the transcribing performance; (2) the validation loss demonstrating the model sensitivity; (3) the spotlight visualization of STN.

5.1 Experimental Setup

5.1.1 Data partition and preprocessing. We partition all our datasets, i.e., *Melody*, *Formula* and *Multi-Line*, into 60%/40%, 70%/30%, 80%/20%, 90%/10% as training/testing sets, respectively, to test model performance at different data sparsity. From each training set, we also sample 10% images as validation set. The images are randomly scaled and cropped for stable training, and ground-truth source code is cut into token sequences in the corresponding language to reduce searching space.

5.1.2 STN setting. We now specify the model setup in STN, including image encoder, transcription decoder and reinforcement module. For STN image encoder, we use a variation of ResNet [11], and set the encoded vector width as 128. For its transcribing decoder, we set the output history embedding h_t, and the spotlight history embedding e_t as the same dimensions of 128, respectively. The value network used at the reinforcement stage is a two-layer fully-connected neural network, with the hidden layer also sized at 128.

5.1.3 Training setting. To set up the training process, we initialize all parameters in STN following [10]. Each parameter is sampled from $U\left(-\sqrt{6/(n_{in} + n_{out})}, \sqrt{6/(n_{in} + n_{out})}\right)$ as their initial values, where n_{in}, n_{out} stands for the number of neurons feeding in and neurons the result is fed to, respectively. Besides, to prevent overfitting, we also add L2-regularization term in the loss function (Equation (5)), with the regularization amount adjusted to the best performance. At reinforcement stage, the discount factor γ is set as 0.99. We also apply some techniques mostly mentioned in [2] to reduce variance, including using an additional target Q-network and reward normalization.

5.1.4 Comparison methods. To demonstrate the effectiveness of STN, we compare our two implementations, i.e., STNM and STNR, with many state-of-the-art baselines as follows.

- **Enc-Dec** is a plain encoder-decoder model used originally for image captioning [33]. Its design allows it to be used in our problem setup with minor adjustments.
- **Attn-Dot** is an encoder-decoder model with attention mechanism following [22], where the attention score is calculated by directly computing the similarity between current output state and each encoded image vectors.
- **Attn-FC** is an encoder-decoder model similar to [33], but with basic visual attention strategy. The model presents two attention strategies, i.e., the "hard" and "soft" attention mechanism, from which we follow [36] and choose the more widely used "soft" attention as it is deterministic and easier to train.

Table 2: Transcription accuracy on three datasets.

(a) **Melody**

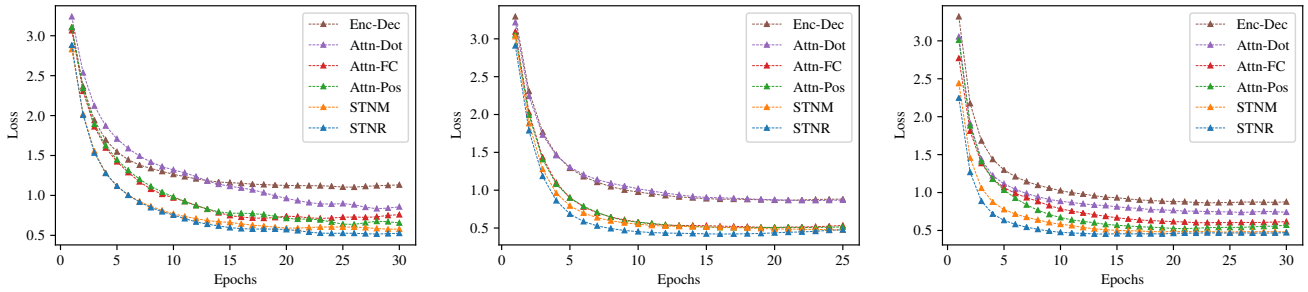
Baseline	Testing set percentage			
	40%	30%	20%	10%
EncDec	0.266	0.272	0.277	0.282
AttnDot	0.524	0.548	0.580	0.617
AttnFC	0.683	0.710	0.730	0.756
AttnPos	0.725	0.736	0.741	0.758
STNM	0.729	0.733	0.749	0.759
STNR	**0.738**	**0.748**	**0.758**	**0.767**

(b) **Formula**

Baseline	Testing set percentage			
	40%	30%	20%	10%
EncDec	0.405	0.427	0.445	0.451
AttnDot	0.530	0.563	0.600	0.611
AttnFC	0.657	0.701	0.717	0.725
AttnPos	0.716	0.723	0.732	0.741
STNM	0.717	0.726	0.740	0.749
STNR	**0.739**	**0.751**	**0.759**	**0.778**

(c) **Multi-Line**

Baseline	Testing set percentage			
	40%	30%	20%	10%
EncDec	0.218	0.227	0.251	0.267
AttnDot	0.334	0.447	0.554	0.599
AttnFC	0.614	0.642	0.686	0.707
AttnPos	0.624	0.652	0.698	0.720
STNM	0.674	0.705	0.731	0.734
STNR	**0.712**	**0.736**	**0.754**	**0.760**



(a) Melody (b) Formula (c) Multi-Line

Figure 6: Validation loss of all models on three datasets.

- **Attn-Pos** is an encoder-decoder model designed specifically for scene text recognition [37], where besides the image content, it also embeds location information into attention calculation, and get superior results.

To conduct a fair comparison, the image encoders for baselines are changed to use the more recent ResNet [11] as our model does, with all of them tuned to have the best performance. All models are implemented by PyTorch[4], and trained on a Linux server with four 2.0GHz Intel Xeon E5-2620 CPUs and a Tesla K20m GPU.

5.2 Experimental Results

5.2.1 Transcribing performance. We train STN along with all the baseline models on four different data partition of each, comparing token accuracy at different data sparsity. We repeat all experiments 5 times and report the average results which are shown in Table 2.

From the results, we can get several observations. First, both STNM and STNR perform better than all the other methods. This indicates that STN framework is more capable for structural image transcription tasks, being more effective and accurate on tracking complex image structures. Second, STN models, as well as attention based methods, all have much higher prediction accuracy than plain EncDec method, which proves the claim mentioned earlier in this paper that image information encoded as a single vector is confounding for decoder to decode, and both STN and attentive models are able to reduce the confusion. Moreover, STN models are consistently better than those attentive ones, showing the superiority of STN with separate modules for spotlighting and transcribing. Third, STNR and STNM has slightly higher performance on *Melody* and *Formula* as Attn-Pos, but surpasses it marginally on *Multi-Line* dataset. These results demonstrate that STN with spotlight mechanism can well preserve the internal structure of images, especially in more complex scenarios, benefiting the transcription accuracy. Last but not least, we can see that STNR consistently outperforms than STNM, which indicates that it is effective to track long-term dependency for spotlighting in the process of transcribing structural image content.

5.2.2 Validation loss. The losses of all models on the validation set throughout the training process on three datasets are shown in Figure 6. There are also similar observations as before, which demonstrates the effectiveness of STN framework again. Clearly, from the results, both STNR and STNM converge faster than the other models, and also achieve a lower loss. Especially, the improvements of them on the more complex Multi-Line datasets are more significant. Thus, we can reach a conclusion that STN with spotlight mechanism has superior ability to transcribe content from structural images. Moreover, all models reach their lowest validation loss before 30 epochs, with STNR and STNM both come to their best point earlier. Thus, in our experiments, we train both STNR and STNM for 25, 15, 20 epochs on Melody, Formula and Multi-Line datasets respectively to obtain the best performance.

[4]http://pytorch.org

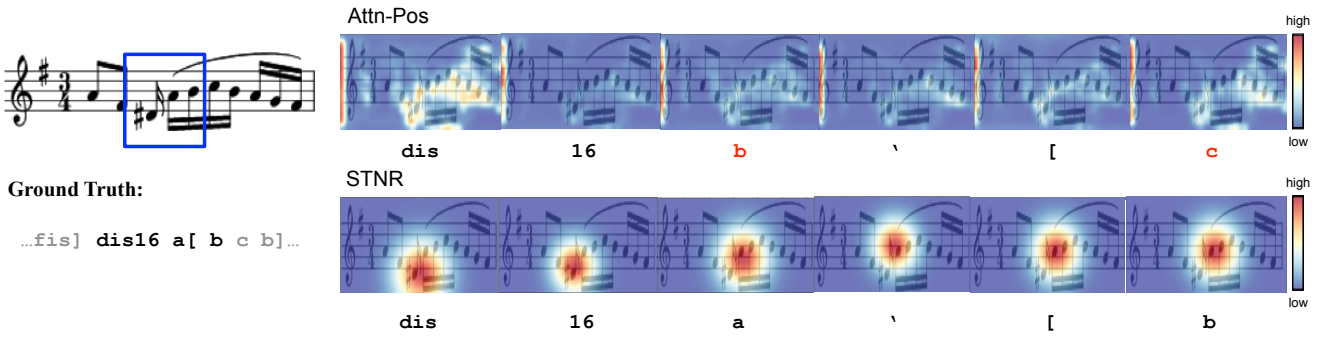


Figure 7: Comparison between attention and spotlight mechanism on Melody dataset.

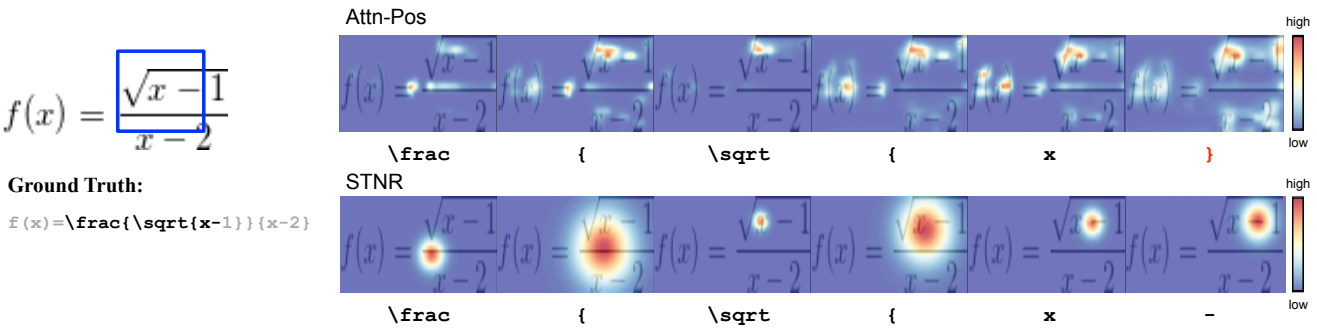


Figure 8: Comparison between attention and spotlight mechanism on Formula dataset.

5.2.3 Spotlight visualization. To show the effectiveness of STN capturing the image structure and producing a reasonable reading path while transcribing, we visualize the spotlight weights computed by STNR when generating tokens, and compare them with the attention weights calculated by Attn-Pos model.

Figure 7 and Figure 8 visualize the results throughout image examples from Melody and Formula datasets, respectively.[5] In each example, we compare the attention and spotlight mechanism on how focused they are when generating a token, also on how well they track the image structure. From the visualization, we can draw conclusions that: (1) STNR finds a more reasonable reading path on both examples. In the melody example, it focuses on notes from left to right, and also tracks the height of each note, making accurate note pitch prediction; In the formula example, it clearly follows middle-top-bottom order when reading a fraction. Attn-Pos model on the other hand, does not track the image structure well enough. As shown in Figure 8, it fails to find the correct spot after generating "`\sqrt{x`", losing track of the radical expression, and generates the wrong token "`}`" at last. (2) Although Attn-Pos model assigns more weights on content objects in images, e.g., notes, formulas and variables, it is often confused at areas with similar content. On the other hand, STNR clearly distinguishes similar regions properly. More specifically, in Figure 7, although Attn-Pos is able to focus on the notes, all notes are given similar weights as they look similar, which causes confusion and then wrong prediction. And in Figure 8, when Attn-Pos writes x, three x's in the image all have high weights, causing the model to forget where to look next. On the contrary, STNR is well focused on the correct spot when generating each token on both of the datasets, which leads to more precise predictions.

5.2.4 Discussion. All the above experiments have shown the effectiveness of STN on structural image transcription tasks. It has superior performance on structural image transcription task compared to other general-purpose approaches, and also captures the structure of the image by producing a reading path following the image structure when transcribing.

There are still some directions for further studies. First, STN learns to transcribe tokens directly with little prior knowledge of the image or specific languages. We are willing to utilize more prior knowledge, such as lexicons and hand-engineered features, to further improve the performance. Second, we will try to apply our model to some more ambitious settings, such as transcribing with long-term context, also to make our model capable for other transcribing applications such as scene text recognition. Third, we would like to further decouple the reading and writing process of STN, in order to mimic human behavior more genuinely.

6 CONCLUSION

In this paper, we presented a novel hierarchical Spotlighted Transcribing Network (STN) for transcribing content from structural

[5]We only choose two real-world datasets for visualization due to the page limitation.

images by finding a reading path tracking the image internal structure. Specifically, we first designed a two-stage "where-to-what" solution with a novel spotlight mechanism dedicated for the "where-to-look" problem, providing two implementations under the framework, modeling the spotlight movement through Markov chain and recurrent dependency, respectively. Then, we applied supervised learning and reinforcement learning methods to accurately train and refine the spotlight modeling, in order to learn a reasonable reading path. Finally, we conducted extensive experiments on one synthetic and two real-world datasets to demonstrate the effectiveness of STN framework with fast model convergence and high performance, and also visualized the learned reading path. We hope this work could lead to more studies in the future.

ACKNOWLEDGEMENTS

This research was partially supported by grants from the National Natural Science Foundation of China (No.s U1605251 and 61727809), the Science Foundation of Ministry of Education of China & China Mobile (No. MCM20170507), and the Youth Innovation Promotion Association of Chinese Academy of Sciences (No. 2014299).

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. A brief survey of deep reinforcement learning. *arXiv preprint arXiv:1708.05866* (2017).
[2] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086* (2016).
[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).
[4] CT Blakemore and Fergus W Campbell. 1969. On the existence of neurones in the human visual system selectively sensitive to the orientation and size of retinal images. *The Journal of physiology* 203, 1 (1969), 237–260.
[5] Yue Cao, Mingsheng Long, Jianmin Wang, Qiang Yang, and Philip S Yu. 2016. Deep visual-semantic hashing for cross-modal retrieval. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1445–1454.
[6] Kam-Fai Chan and Dit-Yan Yeung. 2000. Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition* 3, 1 (2000), 3–15.
[7] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).
[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).
[9] Nobuo Ezaki, Marius Bulacu, and Lambert Schomaker. 2004. Text detection from natural scene images: towards a system for visually impaired persons. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, Vol. 2. IEEE, 683–686.
[10] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. 249–256.
[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition, In Proceedings of the IEEE conference on computer vision and pattern recognition. *CVPR*, 770–778.
[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[13] Zhenya Huang, Qi Liu, Enhong Chen, Hongke Zhao, Mingyong Gao, Si Wei, Yu Su, and Guoping Hu. 2017. Question Difficulty Prediction for READING Problems in Standard Tests. In *AAAI*. 1352–1359.
[14] S Impedovo, L Ottaviano, and S Occhinegro. 1991. Optical character recognition—a survey. *International Journal of Pattern Recognition and Artificial Intelligence* 5, 01n02 (1991), 1–24.
[15] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*. 448–456.
[16] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2016. Reading text in the wild with convolutional neural networks. *IJCV* 116, 1 (2016), 1–20.
[17] Anitha Kannan, Simon Baker, Krishnan Ramnath, Juliet Fiss, Dahua Lin, Lucy Vanderwende, Rizwan Ansary, Ashish Kapoor, Qifa Ke, Matt Uyttendaele, and others. 2014. Mining text snippets for images on the web. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1534–1543.
[18] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[19] Chen-Yu Lee and Simon Osindero. 2016. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2231–2239.
[20] Qi Liu, Runze Wu, Enhong Chen, Guandong Xu, Yu Su, Zhigang Chen, and Guoping Hu. 2018. Fuzzy cognitive diagnosis for modelling examinee performance. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 4 (2018), 48.
[21] Shijian Lu, Linlin Li, and Chew Lim Tan. 2008. Document image retrieval through word shape coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 11 (2008), 1913–1918.
[22] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).
[23] A. Mishra, K. Alahari, and C. V. Jawahar. 2012. Scene Text Recognition using Higher Order Language Priors. In *BMVC*.
[24] Volodymyr Mnih, Nicolas Heess, Alex Graves, and others. 2014. Recurrent models of visual attention, In Advances in neural information processing systems. *NIPS*, 2204–2212.
[25] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.
[26] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 1717–1724.
[27] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* (2015).
[28] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre RS Marcal, Carlos Guedes, and Jaime S Cardoso. 2012. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval* 1, 3 (2012), 173–190.
[29] Shuo Shang, Jiajun Liu, Kun Zhao, Mingrui Yang, Kai Zheng, and Ji-Rong Wen. 2015. Dimension reduction with meta object-groups for efficient image retrieval. *Neurocomputing* 169 (2015), 50–54.
[30] Baoguang Shi, Xiang Bai, and Cong Yao. 2017. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence* 39, 11 (2017), 2298–2304.
[31] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, , Si Wei, and Guoping Hu. 2018. Exercise-Enhanced Sequential Modeling for Student Performance Prediction. In *AAAI*. 2435–2443.
[32] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*.
[33] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator, In Proceedings of the IEEE conference on computer vision and pattern recognition. *CVPR*, 3156–3164.
[34] Kai Wang, Boris Babenko, and Serge Belongie. 2011. End-to-end scene text recognition, In Computer Vision (ICCV), 2011 IEEE International Conference on. *ICCV*, 1457–1464.
[35] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. 2012. End-to-end text recognition with convolutional neural networks, In Pattern Recognition (ICPR), 2012 21st International Conference on. *ICPR*, 3304–3308.
[36] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention, In International Conference on Machine Learning. *ICML*, 2048–2057.
[37] Xiao Yang, Dafang He, Zihan Zhou, Daniel Kifer, and C Lee Giles. 2017. Learning to Read Irregular Text with Attention Mechanisms. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*.
[38] Qixiang Ye and David Doermann. 2015. Text detection and recognition in imagery: A survey. *IEEE transactions on pattern analysis and machine intelligence* 37, 7 (2015), 1480–1500.
[39] Haochao Ying, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*.
[40] Honggang Zhang, Kaili Zhao, Yi-Zhe Song, and Jun Guo. 2013. Text extraction from natural scene image: A survey. *Neurocomputing* 122 (2013), 310–323.