

What You Get Is What You See: A Visual Markup Decompiler

所见即所得：一个 markup 视觉解码系统

作者

Yuntian Deng

Harvard University dengyuntian@gmail.com

Anssi Kanervisto

University of Eastern Finland anssk@student.uef.fi

Alexander M. Rush

Harvard University srush@seas.harvard.edu

Abstart

基于 image caption generation 和光学字符识别的最新进展，我们提出了一种基于深度学习的 解码系统，它能够将图像反编译为标记文本。

虽然这个任务在 OCR 领域中是一个经过充分研究的问题，但我们采用了本质上不同的 data-driven approach 方法。

我们的模型不需要任何关于基础标记语言的知识，并且只需要对现实数据进行 end-to-end 的训练。

该模型使用卷积神经网络进行文本和格式识别，并与基于注意力机制的机器翻译系统相结合。

为了训练和评估模型，我们引入了一个新的数据集，其中包含 来自现实的与LaTeX标记配对的数学公式，还有与HTML片段对应的合成数据集。

实验结果表明，此系统在两个数据集之间生成准确的标记上有让人惊讶的表现。

标准的特定领域 LaTeX OCR 系统可以达到 25% 左右的准确率，而我们的模型精确再现了 75% 用例的渲染图像。

Introduction

Optical character recognition (OCR) 技术一般被用来从图像中识别自然语言；然而，早在 (Anderson 1967) 的工作中，人们对研究如何将图像转化为结构化语言或 markup 有很大兴趣，以定义文本自身和其语义。

这项工作的焦点在数学公式的 OCR 上，以及如何处理减号和上标，特殊符号和分数嵌套的表达等方面 (Belaid and Haton 1984; Chan and Yeung 2000)。

最有效的系统将特殊字符分割与 underlying mathematical layout language 的语法相结合 (Miller and Viola 1998)。

这种方法的一个主要例子是 INFTY 系统，被用于将印刷体数学公式转化为 LaTeX 和其它标记格式 (Suzuki et al. 2003)。

因为这两个领域中深度神经模型的改进，像 OCR 这样需要和图像与文本数据一起处理的问题最近有了新的研究价值。
据。

例如，在手写识别 (Ciresan et al. 2010)，自然场景中的 OCR (Jaderberg et al. 2015; 2016; Wang et al. 2012) 和图像标题生成 (Karpathy and Fei-Fei 2015; Vinyals et al. 2015b) 方面取得了进展。

在 high-level，这些系统都学习输入图像的抽象编码表示，然后进行解码以生成文本输出。

除了在标准任务中表现不错外，这些模型完全是 data driven，这使得它们可以适应各种类型的数据集而不需要大量的预处理数据输入或 domain specific engineering。

转向图像和文本的 data-driven 神经方法引导我们重新审视生成结构化标记的问题。

我们考虑 一个监督模型是否可以学习从图像中生成正确的标记，而不需要基础标记语言的 textual or visual grammar。

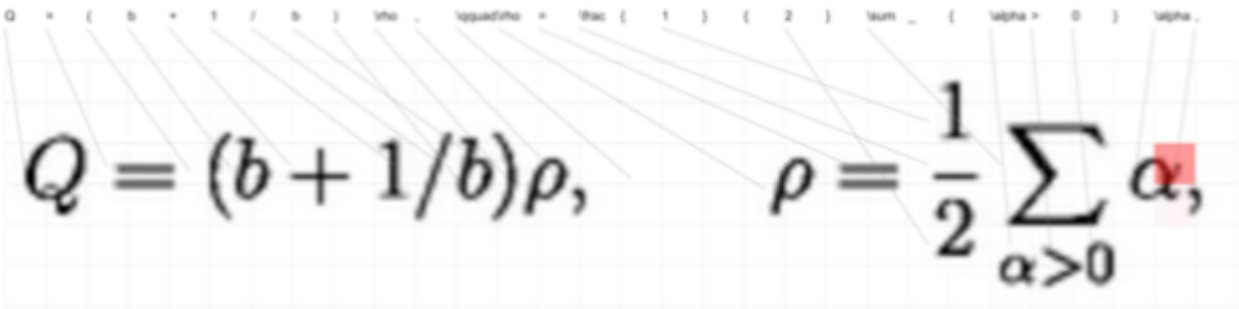
虽然对语言建模的结果表明神经模型始终可以生成句法上正确的标记 (Karpathy, Johnson, and Li 2015; Vinyals et al. 2015a)，还不清楚完整的解决方案是否可以从 markup-image 对中学习。

我们的模型，WYGIWYS [What you get is what you see]，是基于注意力的编码-解码的简单拓展 (Bahdanau, Cho, and Bengio 2014)，现在是机器翻译的标准。

与图像标题中的工作类似 (Xu et al. 2015)，这个模型在图像上结合了多层卷积网络，并使用基于注意力的循环神经网络解码。

为了使该模型适应 OCR 问题并捕获文档的时间布局，我们在应用注意力之前 incorporate a new

注意力的使用额外提供了一种从生成的标记到原始源图像的对齐（见图1）。



（图 1：模型生成数学标记的例子。此模型基于输入图像 x 一次生成一个 LaTeX 符号 y 。灰线突出显示在 CNN V 和 RNN 解码器 V 之后 $H' \times V'$ 的网格特征。虚线表示每个单词 α 的质量中心(仅显示非结构性单词)。红色单元格显示了 the relative attention for the last token。详见：<http://lstm.seas.harvard.edu/latex/> 以获取测试集上此可视化的完整交互式版本。）

我们还为图像标记任务引入了两个新的数据集。

初步试验使用了一个表现为网页的人工合成几何级数 HTML 示例的小型数据集。

对于主要实验，我们加入了新数据集，IM2LATEX-100K，它包含从已发表文章中收集的大量显示世界数学表达式¹。

我们将公开发布此数据作为此工作的一部分。

同样的模型架构被训练用来生成 HTML 和 LaTeX 标记，目标呈现为准确的源图像。

实验将模型的输出与一些研究和 commercial baselines，以及 ablations of the model 进行比较。

完整的数学表达式生成系统能够在图像编辑距离的 15% 内匹配图像，并且在超过 75% 的实际测试中用例中是相同的。

另外，多行编码器的使用使得性能有显著的增加。

所有数据，模型和评估脚本均在 <http://lstm.seas.harvard.edu/latex/> 公开。

Problem: Image-to-Markup Generation

我们将 image-to-markup 问题定义为将渲染的源图像转换为 target presentational markup，该标记完全描述了内容和布局。

源图像， $\mathbf{x} \in \mathbf{X}$ ，是一个具有高度 H 和宽度 W 的图像，例如， $R^{H \times W}$ 表示灰度输入。目标

$\mathbf{y} \in \mathbf{Y}$ ，由一系列 tokens y_1, y_2, \dots, y_C 组成，其中 C 是输出的长度，每一个 y 都是标记语言 vocabulary 集合 Σ 中的标记。

渲染由一个可能未知的，many-to-one, compile function 定义，编译： $\mathbf{Y} \rightarrow \mathbf{X}$ 。

在实践中，这个函数功能可能非常复杂，比如说浏览器，or ill-specified, e.g. the LaTeX language。

The supervised task is to learn to approximately invert the compile function using supervised examples of its behavior.

我们假设我们给出了实例： $(x^{(1)}, y^{(1)}), \dots, (x^{(J)}, y^{(J)})$ ，可能有不同的维度 H, W, C 和 $\text{compile}(y) \approx x$ ，对于所有训练对 (x, y) (假设存在噪声)。

在测试时，系统获得 ground-truth y 渲染的未经处理输入 x 。

它生成一个假设的 y ，然后通过黑盒函数 $x = \text{compile}(y)$ 来呈现。

在 x 和 x 之间进行评估，i.e. 目标是产生类似的渲染图像，而 y 或许不会与 ground-truth 标记 y 相似。

Model

我们的 WYGIWYS 模型对这个任务结合了一些来自视觉和自然语言处理的标准神经组件。

它首先使用卷积神经网络(CNN)提取图像特征，将特征排列在网格中。

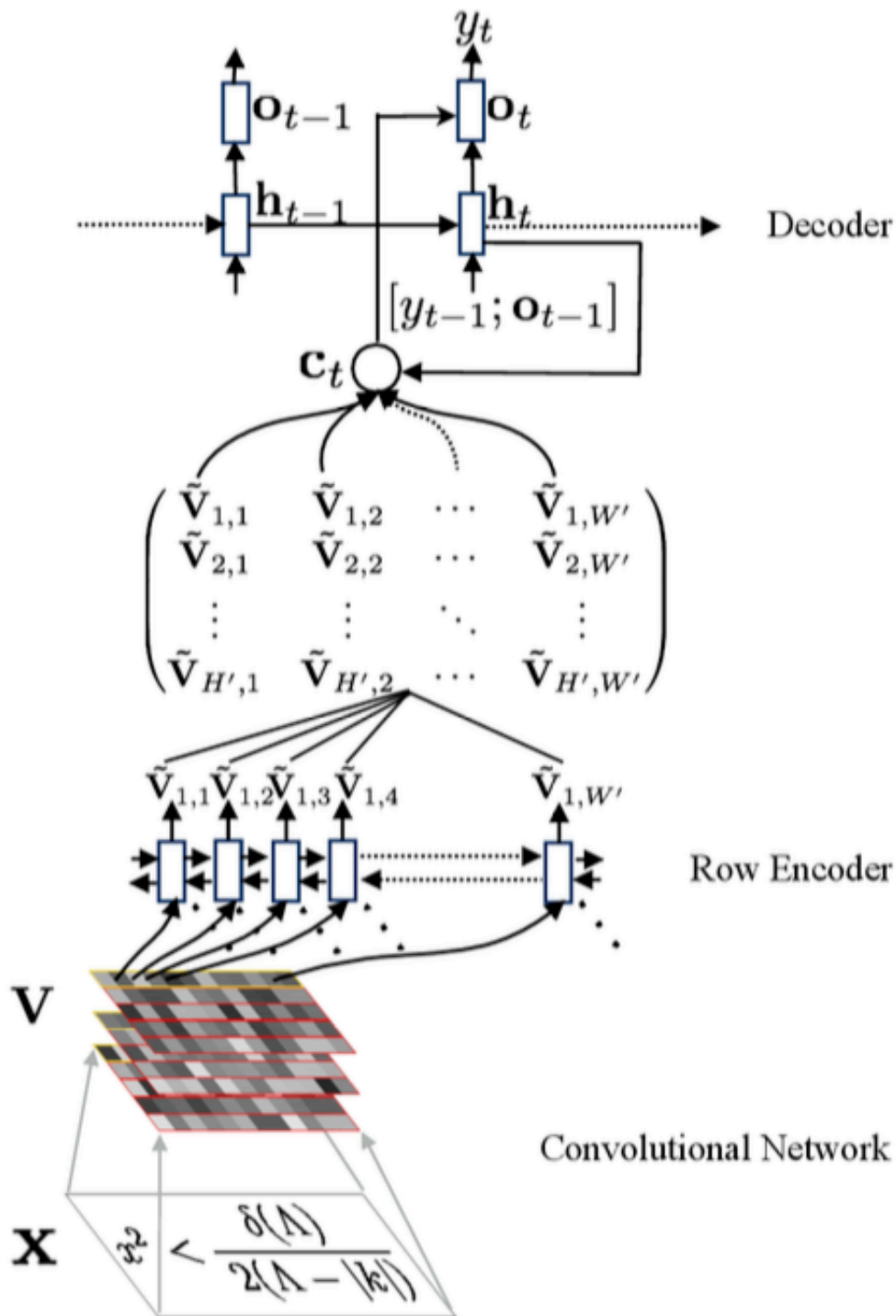
每一行会被循环神经网络(RNN)解码。

这些解码的特征由具有视觉注意力机制的 RNN 编码器使用。

解码器在 vocabulary 集合 Σ 上实现了 conditional language model，并训练完整的模型以最大化观察到的标记的可能性 (the likelihood of the observed markup)。

完整的结构见 图2 的说明。

我们描述了模型的更多细节。



(图 2: WYGIWYS 的网络结构。给定输入图像，一个 CNN 会被用来提取数据特性，然后对于最终特征图中的每一行，我们利用 RNN 解码器解码。被解码的特征由一个具有视觉注意力机制的 RNN 解码器处理，最终输出。为了明确，我们仅展示第一行的 RNN 编码和在时间步骤 t 的解码)

Convolutional Network

一张图片的视觉特性由一个 interleaved with max-pooling layers 的多层卷积神经网络提取。

这个网络架构现在是标准的； we model it specifically after the network used by Shi et al.

(2015) for OCR from images (表2 中给出了明确说明。)

与最近的一些 OCR 研究不同(Jaderberg et al. 2015; Lee and Osindero 2016), 我们没有使用最终完全连接层(Ioffe and Szegedy 2015), 由于我们希望保留 CNN 的 locality 以便使用视觉注意力。

CNN 采用原始输入 $R^{H \times W}$ 和生成大小为 $D \times H' \times W'$ 的特征网络 V , 其中 c 表示 channels 数量, 而 H' 和 W' 是被 pooling 缩小的 (*where c denotes the number of channels and H' and W' are the reduced sizes from pooling*)。

Row Encoder

在基于注意力的图片标题中(Xu et al. 2015), 图像特征网格可以直接传递给解码器。

对于 OCR, 视觉特征 提供给解码器包含重要的相对序列信息。

因此, 我们尝试使用额外的 RNN 编码器模块去重新编码网格中的每一行。

直观地说, 我们希望这样能有这两个方面的帮助: (1)许多标记语言默认 left-to-right order, 编码器可以更容易地学习, (2)RNN 可以利用周围的水平上下文改进隐藏的描述 (*RNN can utilize the surrounding horizontal context to refine the hidden representation*)。

形式上, 循环神经网络(RNN)是一个参数化函数 RNN , 递归地映射输入向量和隐藏的状态到一个新的隐藏的状态。

在时刻 t , 被隐藏的状态由下列一个方式的输入 v_t 更新: $h_t = RNN(h_{t-1}, v_t; \theta)$, h_0 是初始状态。

在实践中, RNN 有许多不同的变体; 然而, 长短期记忆网络(LSTMs)(Hochreiter and Schmidhuber 1997)已经被证明在大多数 NLP 任务中非常有效。

简单起见, 我们会用 RNN 描述这个模型, 但所有实验都使用 LSTM 网络。

在此模型中, 通过在该输入每一行运行一个 RNN, 从 V 创建新的特征网格 V' 。

递归每一行 $h \in \{1, \dots, H'\}$ 和每一列 $w \in \{1, \dots, W'\}$, 新的特征被定义为 $V'_{h,w} = RNN(V_{h,w-1}, V_{h,w})$ 。

为了捕获垂直方向的序列信息, 我们为每一行使用可训练的隐藏初始状态的 $V_{h,0}$, 我们将其命名为位置嵌入(positional embeddings)。

Decoder

接着使用一个基于 sequence of annotation grid V 的解码器生成目标标记令牌 $\{y_t\}$ 。

解码器被训练为一个条件语言模型，以给出下一个已给定历史和注释的下一个令牌的概率。

该语言模型在 RNN 解码器顶部定义，

$$p(y_{t+1}|y_1, \dots, y_t, V) = \text{softmax}(W^{out} o_t)$$

W_{out} 是学习的线性变换， $o_t = \tanh(W^c[h_t; c_t])$ 。向量 h_t 被用来总结解码历史：

$h_t = RNN(h_{t-1}, [y_{t-1}; o_{t-1}])$ 。被定义在下面的上下文向量 c_t 用于从注释网格获取上下文信息。

在每一个时间 t ，上下文向量 c_t 会考虑注释网格。然而，由于很多注释单元可能不相关，模型需要知道哪些单元需要注意。

我们使用一个注意力模型 $a(\cdot)$ 来对这种 alignment 建模(Bahdanau, Cho, and Bengio 2014)：

$$e_t = a(h_t, \{V_{h,w}\})$$

$$\alpha_t = \text{softmax}(e_t)$$

$$c_t = \phi(\{V_{h,w}, \alpha_t\})$$

α_t 是基于 e_t 计算的权重，权重向量 α_t 和所有的注释向量 $\{V_t\}$ 都被组合成上下文向量 c_t 。

注意这里 α 和 ϕ 有不同的选项，我们遵循过去工作的经验，使用

$$e_{it} = \beta^T \tanh(W_h h^{i-1} + W_v v_t) \text{ 与 } c_i = \sum_t \alpha_{it} v_t \text{ (Luong, Pham, and Manning 2015)}。$$

向量 c_t 和 h_t 被联系在一起预测令牌 y_t 的概率。

图1 显示了一个模型中每一步注意力分布的真实例子。

Training and Generation

完整的模型被 end-to-end 地训练，以最大化观察到被训练数据的可能性。

除了训练数据之外，模型不会被给予任何关于标记语言或生成过程的其他信息。

为了从未知图片生成标记，我们简单的使用 beam search 在测试时，没有进一步的硬约束。

Data

有一些数据集可以用于图像到标记的生成问题(Mouchere et al. 2012; 2013; Lin et al. 2012)，他们对于训练一个 data-driven 系统而言，都太小了。

我们因此为这个任务引入了两个新的数据集：一个有 HTML 片段的初步网页数据集，和一个大型实际 LaTeX 数学公式数据集。

Web Page-to-HTML

我们的初步数据集是 a synthetically generated collection of “web pages”，用于测试模型是否可以学习相对空间位置。

我们生成一个由 100K 个唯一 HTML 片段和相应的渲染图片组成的数据集。

图像由 WebKit 编译生成大小为 100×100 的图像。

任务是根据渲染的图像推出 HTML 标记。

HTML 标记使用简单的上下文无关法确定。

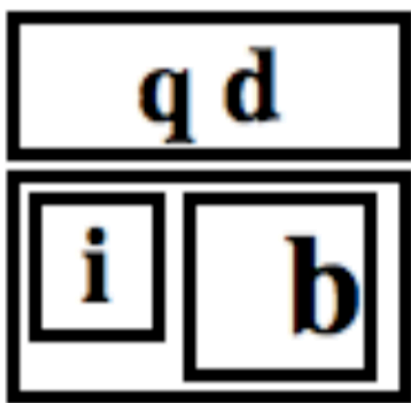
该语法递归地生成 div，每一个 div 都有实线边界，随机的宽度和随机的 float(左或右)。

每一个 div 可以选择垂直或水平地分割。²

递归的最大深度限制为两个嵌套的 div 层。

最后我们为元素宽度添加了一些限制以避免重叠。

图3展示了一个从数据集中取样的示例片段。



```
<div style=' margin:2px; border:
solid; text-align:center; ' > <
span style=' font-weight:bold;
text-align:left; font-size:150%; '
> q d</span> </div> <div style='
margin:2px; border:solid; ' > ...
```

(图3：示例网页图像和它相应的 HTML 片段。)

Math-to-Latex

我们的主要数据集，IM2LATEX-100K，收集了大量用 LaTeX 写的真实数学公式。

这个数据集为学习如何重现自然存在的标记提供了更困难的 test-bed。

Corpus

IM2LATEX-100K 数据集提供 103556 种不同的 LaTeX 数学方程式和它们的渲染图像。

我们通过解析 arXiv 上论文中的 LaTeX 资源获取公式。

LaTeX sources are obtained from tasks I and II of the 2003 KDD cup (Gehrke, Ginsparg, and Klein-berg 2003) containing over 60,000 papers.

为了从 LaTeX 资源中提取出公式，我们匹配 LaTeX 资源，我们使用了简单的正则表达式，即 `\begin{equation} (.*?)\end{equation}`，`$$ (.*?) $$`，`\[(.*?) \]` 和 `\((.*?) \)`。

我们仅保留匹配到的字符总数在 40 到 1024 之间的结果以避免使用单个符号，大型 matrices 或文本段。

通过这些设置，我们提取到了超过 8000,000 中不同的 LaTeX 公式。

在剩余的公式中，大概十万个在 vanilla LaTeX 环境中渲染。

公式的渲染用 pdf`latex`³ 实现，排除了无法编译的公式。

被渲染的 PDF 文件被转换为 PNG 格式⁴。

我们提供的最终数据集提供了 103,556 张 resolution 为 1654×2339 的图像，with black equations against transparent background，以及相应的 LaTeX 公式。

数据集分为训练集(83,883 个等式)，验证集(83,883 个等式)和测试集(9,319 个等式)，用于标准化测试步骤。

LaTeX 公式的范围在 38 到 997 个字符之间，平均值为 118，中位数是 98。

Tokenization

训练模型需要设置令牌集 Σ 。

一种选择是使用纯粹基于字符的模型。

虽然这种模型需要少量的假设，但基于字符的 NMT 模型比基于字的模型效率更低，更慢，要使用大量的内存。

因此，原始标记被简单地分为有意义的最小 LaTeX tokens，比如说对观察到的字符，符号如 `\sigma`，修饰符 `^`，函数，强调，环境，括号和其他各式各样的命令。

Optional: Normalization

最后我们注意到，自然生成的 LaTeX 中，很多不同的表达式生成了相同的输出。

因此，我们尝试使用一个可选的 normalization 预处理步骤以排除虚假的歧义(在训练之前)。

为了 normalization，我们写了一个 LaTeX 解析器⁵来把标记转换为抽象语法树。

我们接着应用一组安全的规范化树变换来消除常见的虚假歧义，如表1所示。

注意这只会改变训练数据，不会改变模型本身。

Transform	Original	Normalized
SubSup	H^I_1	H_I^1
ReqBrack	H^I	$H^{\{I\}}$
Desugar	H'	$H^{\{\backslashprime\}}$
ExpOperators	\sin	sin
InfixPrefix	\over	$\frac{\{\}\{\}}$
MatrixEnv	\matrix	$\begin{array}\dots$
Drop	$\label{\}$	$-$

(表1：在规范化模式下应用于 LaTeX 抽象语法树的预处理转换。这些转换大多是安全的，尽管一些极端情况会对输出有微小影响。)

Experimental Setup

为了测试这种方法，我们将所提出的模型与一些经典 OCR baselines，神经模型和 ablations 在 HTML 与 LaTeX 反编译任务上进行比较。

Baselines

目前最好的基于 OCR 的数学表达式识别系统是 InftyReader 系统，是 INFTY 系统的专有商业实现(Suzuki et al. 2003)。

这个系统整合了符号识别和结构分析阶段。

另外，我们尝试使用开源 AbiWord OCR 系统，该系统包含了一个 Tex 生成模式(Wen 2002)。

然而我们发现这个系统在这项任务上表现太差，难以进行比较。

对于神经模型，a natural comparison is to standard image captioning approaches (Xu et al. 2015)。

由于我们的模型基于这种方法，我们通过移除 RNN 编码器进行比较，即用 V 替换 V ，并增加 CNN 的数量，使参数的数量大致相同。

我们将此模式命名为 textscCNNEnc。

我们还进行了此模型与传统 LM 方法的比较，包括一个标准 NGRAM 模型(5-gram trained with Kneser-Ney smoothing)和 LSTM-LM。

这表明提高多少取决于对基础语言标记建模的改进。

最后，对于 LaTeX，我们也评估了完全归一化，norm, versus simple tokenization, tok。

Evaluation

我们的核心评估方法是检查输出的渲染标记图像 \hat{x} 与正确的图像 x 比较得出的准确性。

主要评估方法反映了在 gold 与 预测图像之间的 the column-wise edit distance。

我们明确地离散化生成的列，并比较 the edit distance sequences。

最终得分是 edit distance ops used 的总数除以数据集的最大数量。

此外，我们检查了原始图像的精确匹配准确度还有消除空列后的值。⁶

我们也加入了标准化的 intrinsic text generation metrics, conditional language model perplexity 和 BLEU 得分(Papineni et al. 2002)。

要注意这两个指标都对标记语言有虚假歧义的事实很敏感，所以 1 的 deterministic perplexity 是不可能的。

Implementation Details

同样的模型和超参数被用来做 image-to-markup 任务。

CNN 规范总结在表2中。

Conv	Pool
c:512, k:(3,3), s:(1,1), p:(0,0), bn	-
c:512, k:(3,3), s:(1,1), p:(1,1), bn	po:(1,2), s:(1,2), p:(0,0)
c:256, k:(3,3), s:(1,1), p:(1,1)	po:(2,1), s:(2,1), p(0,0)
c:256, k:(3,3), s:(1,1), p:(1,1), bn	-
c:128, k:(3,3), s:(1,1), p:(1,1)	po:(2,2), s:(2,2), p:(0,0)
c:64, k:(3,3), s:(1,1), p:(1,1)	po:(2,2), s:(2,2), p(2,2)

(Table 2: CNN specification. ‘Conv’: convolution layer, ‘Pool: max-pooling layer. ‘c’: number of filters, ‘k’: kernel size, ‘s’: stride size, ‘p’: padding size, ‘po’: , ‘bn’: with batch normalization. The sizes are in order (height, width).)

模型对所有 RNN 使用单层 LSTM。

我们为编码器使用了双向 RNN。

RNN 编码器的隐藏状态大小为 256，解码器 RNN 的大小为 512，token 嵌入大小为 80。

模型总共有 948万 个参数。

我们使用小批量随机梯度下降(mini-batch stochastic gradient descent)来学习参数。

初始学习速率被设置为 0.1，一旦验证困惑度没有降低我们会将其减半。

我们训练这个模型 12 epochs，并使用验证困惑度来选择最优模型。

在测试阶段，我们使用 beam 大小为 5 的 beam search。

这个系统使用基于 Seq2seq-attn NMT 系统的 Torch (Collobert, Kavukcuoglu, and Farabet 2011)构建⁷。

实验运行在 12GB NVidia Titan X GPU。

HTML

所有的图像都以 100×100 个颜色输入开始，接着它们会被 down-sampled 为 64×64 的灰度图。

我们接着将像素归一化到 $[-1, 1]$ 区间中。

在训练时，我们仅使用少于 100 个 token 输出的训练实例，以加速训练过程。

批量大小被设置为 100.

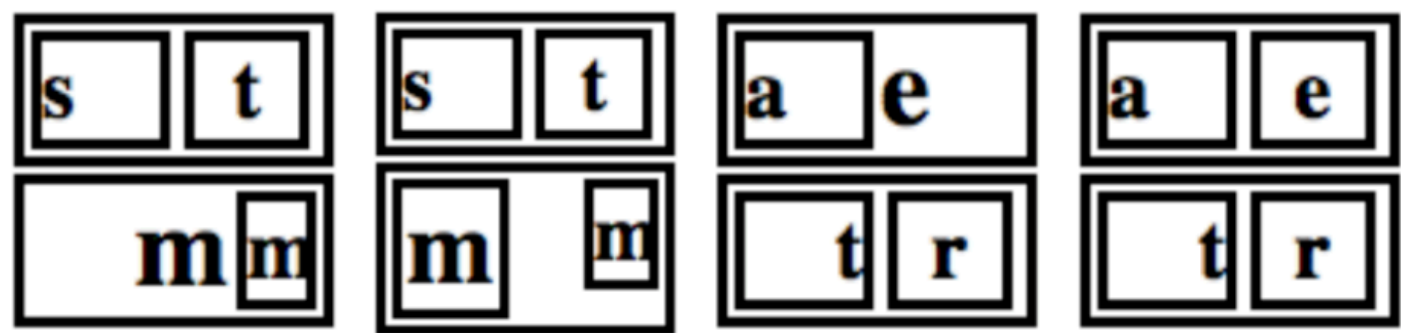
这个训练过程花了 4 小时。

LaTeX

原始图像被裁减为仅剩公式区域，并在四周留出了 8 像素的 padded。

为了提高效率，我们 downsample 所有图片到原始大小的一半。

为了便于批量处理，we group images into similar sizes 并用空白填充。⁸



(Figure 4: Typical errors in the HTML dataset. Left to right: 1st ground truth, 1st predicted, 2nd ground truth, 2nd predicted.)

在训练和测试期间，忽略所有的大尺寸图片，多于 150 个token 的 LaTeX 公式，或不能被我们的解析器解析的图片(但在测试中包括)。

由于 GPU 内存的限制，训练批量大小设置为 20。

训练过程花了 20 小时。

Results

HTML 数据集的初始实验如表4所示。

Model	Train Perp	Test Perp	Exact Match
NGRAM	1.60	1.60	-
WYGIWYS	1.06	1.06	97.61

(Table 4: Web Page-to-HTML results. Reports the training perplexity, test perplexity, as well as

exact match score on test set.)

该模型能够达到 1.06 的困惑度，而且准确匹配精度超过 97.5%。

这些结果表明该模型能够基于 spatial cues 学习识别和生成正确的输出。

大多数剩余的困惑是由于底层标记语言的歧义。

典型的错误如表4所示。

少数问题出现在 font-size 和 div 的相对大小上。

数学表达式的主要实验结果在表3中给出。

Model	Preprocessing	BLEU (tok)	BLEU (norm)	Edit Distance	Exact Match	Exact Match (-ws)
INFTY	-	51.20	66.65	53.82	15.60	26.66
CNNENC	norm	52.53	75.01	61.17	53.53	55.72
WYGIWYS	tok	73.71	73.97	84.26	74.46	77.04
WYGIWYS	norm	58.41	87.73	87.60	77.46	79.88

(Table 3: Main experimental results on the IM2LATEX-100K dataset. Reports the BLEU score compared to the tokenized formulas (BLEU (tok)), and the BLEU score compared to the normalized formulas (BLEU (norm)), column-wise image edit distance, exact match, and exact match without whitespace columns.)

这些结果在反编译已被渲染的 LaTeX 任务上对几个不同的系统进行了比较。

经典的 INFTY 系统在文本准确性方面表现相当不错，但在更严格的图像指标上表现不佳。

我们对这个 Image-to-Caption 工作的 CNNENC 重新实现做得更好，将数字提高到50%。(Our reimplementa-tion of the Image-to-Caption work CNNENC does much better, pushing that number to above 50%.)

我们的完整系统使用 RNN 编码器将这个值提高到75%以上，在此任务上取得了非常高的精度。

我们期望 LaTeX normalizer 能够大幅度提高性能，尽管实现了高 normalized BLEU。但只能提供几点精度增益。

这表明解码器 LM 能够很好地学习实际 LaTeX 中的歧义。

为了更好地理解模型各部分的贡献，我们进行了消除不同方面的消融实验，如表5所示。

Model	Ablation	Train Perp	Test Perp
NGRAM		5.50	8.95
LSTM-LM	-RNNEnc-CNN	4.13	5.22
CNNENC	-RNNEnc	1.08	1.18
WYGIWYS	-PosEmbed	1.03	1.12
WYGIWYS		1.05	1.11

(Table 5: Image-to-LaTeX ablation experiments. Compares simple LM approaches and versions of the full model with different encoders.)

最简单的模型是 LaTeX 上的标准NGRAM LM，困惑度大约为 8。

简单地切换到 LSTM-LM 将值降低到 5，可能是因为它能够计算括号和嵌套级别。

使用 CNN 添加图像数据进一步将困惑降低至 1.18。

添加编码器 LSTM 会将减小添加一点到 1.12，但实际上会在最终精度上产生很大差异。

添加位置嵌入(每行的可训练初始状态)会增加微小的增益。

表6 中显示了模型的主要 non-spacing errors。

<code>\mathcal</code>	\Rightarrow	<code>-</code>	<code>\,</code>	<code>(</code>	\Rightarrow	<code>\left(</code>
<code>\right)</code>	\Rightarrow	<code>)</code>	<code>\left[</code>		\Rightarrow	<code>[</code>
<code>\left(</code>	\Rightarrow	<code>(</code>	<code>\left(</code>		\Rightarrow	<code>\,</code>
<code>)</code>	\Rightarrow	<code>\right)</code>	<code>\big</code>		\Rightarrow	<code>-</code>
<code>\mathbf</code>	\Rightarrow	<code>-</code>	<code>\right)</code>		\Rightarrow	<code>) \,</code>
<code>(</code>	\Rightarrow	<code>\left(</code>	<code>\prime</code>		\Rightarrow	<code>-</code>
<code>\mathrm</code>	\Rightarrow	<code>-</code>	<code>_ {</code>		\Rightarrow	<code>-</code>
<code>\right]</code>	\Rightarrow	<code>]</code>	<code>\widetilde</code>		\Rightarrow	<code>-</code>
<code>\mathrm</code>	\Rightarrow		<code>\bot</code>		\Rightarrow	<code>\perp</code>
		<code>\operatorname</code>				

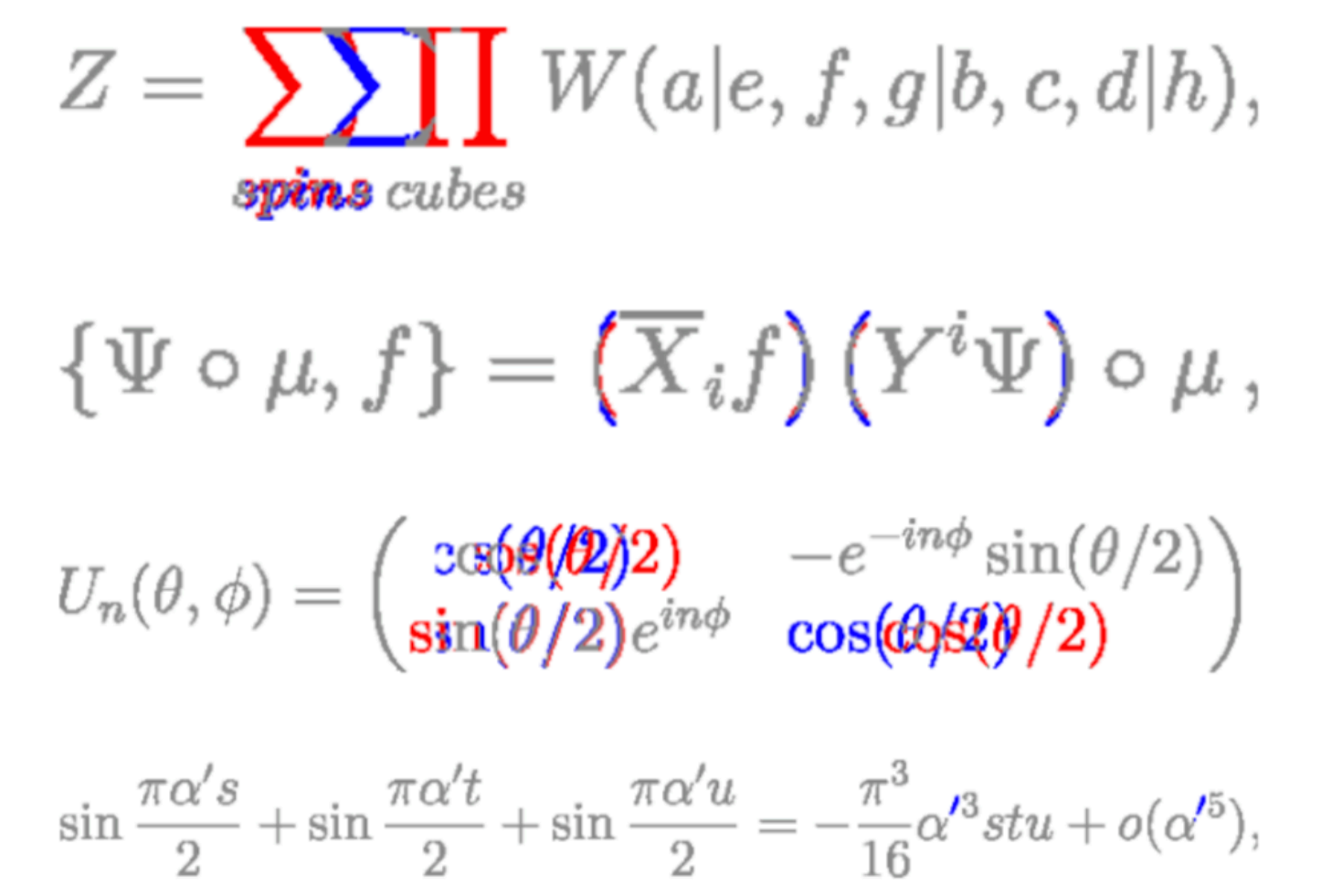
(Table 6: Most common presentation-affecting errors in the LaTeX norm validation dataset.)

这些结果显示了最常见的 presentation-affecting replacements of the norm model。

大多数错误依旧来自字体问题，例如使用小括号而不是大括号，或使用标准数学字体 escaping or using math cal。

图5 显示了常见错误的几个示例。

通常，表达式的大多数结构都会被保留，但会出现一个或两个符号识别错误。



(Figure 5: Typical errors in the LaTeX dataset. We show the operations needed to get ground truth from the rendered pre- dictions. Red denotes add operations and blue denotes delete operations.)

Conclusion and Future Work

我们描述了一个基于视觉注意的模型，WYGIWYS，用于表示标记的 OCR。

对像 HTML 和 LaTeX 来说，这个模型扮演了“可视反编译器”的角色。

我们也引入了新的数据集 IM2LATEX-100K，它为 Image-to-Markup 这项任务提供了一个 test bed。

这些贡献提供了一个在结构化文本 OCR 任务的新观点，并展示了即使没有任何关于语言的基础知识， data-driven 模型也令人惊讶地有效。

这项语言在未来可能的方向有：拓展系统以部署在完整网站上或用来反编译文档，用类似方法处

理手写数学公式或来自草稿的 HTML，或将这些方法与神经推理机，比如说MemNNs(Weston, Chopra, and Bordes 2014)结合起来，用于更复杂的标记或 reference variables。

References

- Anderson, R. H. 1967. Syntax-directed recognition of handprinted two-dimensional mathematics. In Symposium on Interactive Systems for Experimental Applied Mathematics: Proceedings of the Association for Computing Machinery Inc. Symposium, 436–459. ACM.
- Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
- Belaid, A., and Haton, J.-P. 1984. A syntactic approach for handwritten mathematical formula recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1):105–111.
- Chan, K., and Yeung, D. 2000. Mathematical expression recognition: a survey. *IJDAR* 3(1):3–15.
- Ciresan, D. C.; Meier, U.; Gambardella, L. M.; and Schmidhuber, J. 2010. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation* 22(12):3207–3220.
- Collobert, R.; Kavukcuoglu, K.; and Farabet, C. 2011. Torch7: A matlab-like environment for machine learning. In BigLearn, NIPS Workshop, number EPFL-CONF-192376.
- Gehrke, J.; Ginsparg, P.; and Kleinberg, J. 2003. Overview of the 2003 kdd cup. *ACM SIGKDD Explorations Newsletter* 5(2):149–151.
- Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.
- Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of The 32nd International Conference on Machine Learning, 448–456.
- Jaderberg, M.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2015. Deep structured output learning for unconstrained text recognition. *ICLR*.
- Jaderberg, M.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2016. Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision* 116(1):1–20.
- Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3128–3137.
- Karpathy, A.; Johnson, J.; and Li, F.-F. 2015. Visualizing and understanding recurrent networks. arXiv preprint arXiv:1506.02078.
- Lee, C.-Y., and Osindero, S. 2016. Recursive recurrent nets with attention modeling for ocr in the wild. arXiv preprint arXiv:1603.03101.
- Lin, X.; Gao, L.; Tang, Z.; Lin, X.; and Hu, X. 2012. Performance evaluation of mathematical formula identification. In Document Analysis Systems (DAS), 2012 10th IAPR International

Workshop on, 287–291. IEEE.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. EMNLP.

Miller, E. G., and Viola, P. A. 1998. Ambiguity and constraint in mathematical expression recognition. In AAAI/IAAI, 784–791.

Mouchere, H.; Viard-Gaudin, C.; Kim, D. H.; Kim, J. H.; and Garain, U. 2012. Icfhr 2012 competition on recognition of on-line mathematical expressions (crohme 2012). In Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on, 811–816. IEEE.

Mouchere, H.; Viard-Gaudin, C.; Zanibbi, R.; Garain, U.; Kim, D. H.; and Kim, J. H. 2013. Icdar 2013 crohme: Third international competition on recognition of online handwritten mathematical expressions. In 2013 12th International Conference on Document Analysis and Recognition, 1428–1432. IEEE.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics, 311–318. Association for Computational Linguistics.

Shi, B.; Bai, X.; and Yao, C. 2015. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. arXiv preprint arXiv:1507.05717.

Suzuki, M.; Tamari, F.; Fukuda, R.; Uchida, S.; and Kana-hori, T. 2003. Infty: an integrated ocr system for mathematical documents. In Proceedings of the 2003 ACM symposium on Document engineering, 95–104. ACM.

Vinyals, O.; Kaiser, Ł.; Koo, T.; Petrov, S.; Sutskever, I.; and Hinton, G. 2015a. Grammar as a foreign language. In Advances in Neural Information Processing Systems, 2755–2763.

Vinyals, O.; Toshev, A.; Bengio, S.; and Erhan, D. 2015b. Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3156–3164.

Wang, T.; Wu, D. J.; Coates, A.; and Ng, A. Y. 2012. End-to-end text recognition with convolutional neural networks. In Pattern Recognition (ICPR), 2012 21st International Conference on, 3304–3308. IEEE.

Wen, H. 2002. Abiword: Open source’s answer to microsoft word. Linux Dev Center, downloaded from <http://www.linuxdevcenter.com/lpt/a/1636> 1–3.

Weston, J.; Chopra, S.; and Bordes, A. 2014. Memory networks. arXiv preprint arXiv:1410.3916.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In Proceedings of The 32nd International Conference on Machine Learning, 2048–2057.

1. This dataset is based on the challenge originally proposed as an OpenAI Request for

Research under the title Im2Latex. ↩

2. Additionally each div contains a randomly generated span of characters of a random font size. ↩
3. LaTeX (version 3.1415926-2.5-1.40.14) ↩
4. We use the ImageMagick convert tool with parameters -density 200 -quality 100 ↩
5. Based on KaTeX parser <https://khan.github.io/KaTeX/> ↩
6. In practice we found that the LaTeX renderer often misaligns identical expressions by several pixels. To correct for this, only misalignments of ≥ 5 pixels wide are “exact” match errors. ↩
7. <https://github.com/harvardnlp/seq2seq-attn> ↩
8. Width-Height groups used are (120,50), (160,40), (200,40), (200,50), (240,40), (240,50), (280,40), (280,50), (320,40), (320,50), (360,40), (360,50), (360,60), (360, 100), (400,50), (400,160), (500,100)。 ↩