2019 年全国大学生信息安全竞赛
作品报告

作品名称：　基于深度学习的抗屏摄文档水印技术

电子邮箱：　hustcw1998@gmail.com

提交日期：　2019 年 6 月 4 日

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用 A4 纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5 倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。（本页不删除）

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

目录

摘要

随着商业化进程加速，人们越来越重视个人知识产权的保护。但是成果的窃取盗用事件频频发生，这严重伤害了作者本身的利益。现在越来越多的盗取者使用拍照的形式盗取他人成果，并将盗取的成果发布或转卖来谋取利益。例如在公司内部出现泄密者，将重要文件或展示在屏幕上的技术方案拍照泄露。因此我们需要一种技术，能从盗取者拍摄的照片中提取出水印信息来进行泄密溯源。然而，因为屏摄引起的失真、摩尔纹、抖动等因素的影响，传统的文档水印技术已经不再适用。

针对这种情况，本作品提出了一种具有高不可见性、高鲁棒性且抗屏摄的中文文档水印方案。我们通过字体的不同来编码信息，同时训练了一个神经网络用于区分字体，以此提取文档中水印信息。需要注意的是，选择的字体应该为肉眼不可区分的，否则盗取者可能会发现并去掉水印。我们选择了视觉上不可区分的两类字体，并通过众包技术进行相似度衡量，以确保文字的相似性，从而实现编码消息的不可见性。

我们还实现了一个原型系统来演示效果，系统由两个模块组成：文档生成模块和信息恢复模块。文档生成模块分为水印信息编码和文档生成两个部分。将水印信息进行 BCH 编码后，根据‘0/1’比特串产生含有水印信息的文档。信息恢复模块分为文档矫正、切分字体、识别字体、信息解码四个部分。用户将在盗取者拍摄的照片进行校正，并按序切分成单个字体图片。将字体图片输入分类网络得到比特串，再经过 BCH 解码就能恢复原始的水印信息。

系统演示了我们的算法对屏摄泄密文档的溯源能力，但算法的能力不仅限于抗屏摄，对其他泄密方式也具有很强的溯源能力。例如，文档被打印盗取。我们将文档扫描成图片进行字体识别，这样就能得到打印文档中水印信息。对于直接拷贝的电子文档，提取出电子文档中的文字进行识别也可以恢复出水印信息。

第一章　作品概述

第一节　研究背景

随着信息化时代的到来，电子文档成为日常办公文档的主要形式。目前对于重要文件的泄密问题主要还是依靠传统的方案来解决，通过加密，访问控制，身份认证等手段对电子文档进行保护。但是随着智能手机的普及，文档的泄密仅需在屏幕上展示文档，用手机拍摄文档两步即可完成，且拍照的行为本身不易被监控，所以我们需要在文档中嵌入水印来保证文档的可溯源性，从而保护文档的知识产权。传统的文档水印主要针对图像处理过程，在经过屏摄信道后，嵌入的消息很难被无损的恢复，也就无法保证屏摄泄密的可溯源性。这使得设计一种可以抵抗屏摄过程的文档水印方案势在必行。

针对文档的抗屏摄水印技术可以将屏幕编号和时间戳嵌入到文档中。在文件因拍照泄漏后，我们可以在屏摄图片中提取出嵌入的设备编号和时间戳信息找到泄密的设备及泄密时间，从而缩小泄密的排查范围，起到溯源的作用。因此，研究此项技术有着迫切的需求与极大的价值。

本算法以中文这一广泛使用的象形文字为基础进行嵌入设计，不同的字形表示了不同的比特的消息，通过选择不同字形的字体，实现消息的嵌入。即使经过了屏摄处理之后，象形文字的字体信息仍然可以保留。为了实现水印的不可见性，文档应用肉眼不可区分的多种字体构成。准确识别本身非常相近的字体是一个难点，特别是在经过屏摄处理之后。因为屏摄信道会使字形信息产生大量失真。这一问题难以使用传统的算法进行解决，因此本作品使用深度学习技术构建了学习能力足够强的神经网络，同时拍摄了大量的训练数据来增强网络的泛化能力。结合识别字体的神经网络和带有纠错能力的编码方案，本系统实现的字体编码文档水印技术可以大幅增强抗屏摄能力。

第二节　相关工作

我们的工作基于 Chang 等人 [1] 提出的利用英文字体的差别进行水印嵌入工作。相比于他们的工作，我们有以下三个方面改进。第一个方面，选择的语言不同。英文字母只有 26 个，然而我们进行的中文字水印需要识别的常用字就有 3000 及以上。这对我们的神经网络识别能力提出了非常高的要求，在采集训练集时数据量也会大大增加。第二个方面，中文字体和英文在字形上有很大不同。考虑到中文字体有偏旁部首（整个字体不是连通的），分割取字也变得更有难度。第三个方面，我们是对屏幕上显示的文档拍照进行识别。在屏摄过程中

产生的失真、摩尔纹、噪点等都会大大影响神经网络模型的识别率。

编码字体的选择 我们的字体编码工作需要选择字形相似且不易被人察觉出差异的字体。目前有很多种选择字体的方法。近来，众包技术 [2] 以及提取相似度 [3] 经常被使用。考虑到我们的系统只需要验证方案的可行性。我们在之前的已有的技术上做了改进，只选择部分可行字体进行实现，并使用众包技术验证字体的不可区分性。

支持拍照后的字体识别 由于拍照会导致原本文档中的信息流失，大部分的水印技术都无法有效的抵抗屏摄信道带来的失真。Jiren 等人 [4] 提出了一种能够抵抗屏摄过程的图像水印算法，这与我们拍照后识别字体具有很大的相似度。但是，图像水印与字体水印还是有本质的差异。图像水印可以将水印信息置于整张图片中（大部分是一个规则的矩形），水印面积更大，也更易识别水印区域。我们的字体水印识别别区域仅在字体本身，水印面积很小，每个字体的识别区域也不同。

之前也有很多识别图片中字体的工作，如 Carlos[5] 等人提出的使用静态方法提取字体特征进行识别。进来，神经网络与深度学习技术崛起兴盛，使用神经网络识别字体的技术越来越多。Chen 等人 [6] 就提出了一种规模可变的监督学习方法。随后 Wang 等人 [7] 提出使用 CNN 识别字体的方案。我们的识别方法也运用了深度学习技术。基于 ResNet18[8]，我们调整了模型的部分细节和参数以适应于整个任务。

在文档中添加水印信息 我们的工作与水印技术相关。各种载体上（图像、光照、音频、视频）的水印技术都被研究了多年。但是基于文档中文字的水印更有挑战性，目前较为成熟的水印方案也比较少。我们在这里列出几种现有的文档中水印信息的方案。

第一种方案是 wayner 等人 [9][10] 提出的 cover text generation CTG。这种方案生成一种语法和句法上非常自然的文档，在词句之中隐藏信息。这种方案的缺点也是非常明显的——它不能给定文档添加水印。

第二种方案是黄兴等人 [11] 的工作，将字体边缘的像素翻转来进行编码。这种方案的缺点在于信息流失的可能性太大，特别是在屏摄后能识别出水印信息的概率非常小。而且，如果采用这种方法，对所有格式文档适应的成本也非常高。

第三种方案利用了文档格式上的特征，例如改变文字大小、颜色、下划线等特征 [12]。这种方案可以大大提高识别的准确度，但是没有保证水印的不可见性。

水印信息的编码解码 由于编码所采用的字体差别非常小，在屏摄过后字体差别会更小，神经网会有一定概率识别出错。在这种情况下，使用纠错码是非常有必要的。我们使用了 BCH(127,50)[13]。该编码算法能允许 48 比特内出 5 比特错, 即网络允许的 BER 为 10%。只要我们的神经网络识别错误率在 BCH 编码允许内，就能完整的还原水印的信息。最终，我们的网络确实达到了要求，能够还原出水印信息。

第三节 特色描述

第一小节 作品特色

本作品实现了抗屏摄的文档字体水印技术。随着照相技术的普及，抗屏摄已经成为水印技术的重大难点。传统水印处理过的文档在通过屏摄、拍照处理后，会出现严重的失真，无法还原出正确的水印信息。针对现有算法无法解决的这一难点问题，本作品创新性的使用了两种肉眼不可区分的字体作为作为信息载体表达不同消息，通过预编码的消息选择对应的字体生成文档从而实现消息的嵌入。而在水印提取端，通过深度学习训练分类网络来实现不同字体的识别与分类，完成消息的解码。由于深度学习能学到更高级的特征，这意味着在失真严重的屏摄图片中，运用神经网络仍然能提取出分类器所需要的特征进行信息的分类与还原。因此本作品的水印方案能够很好地解决屏摄后信息丢失的问题，即无损地从手机拍摄的水印文档的图片中还原出原始信息。

第二小节 核心算法特色

深度学习数据集是深度学习最为重要的组成部分，我们分析了多种字符之间的相似性，并选定了两种字符集进行多角度拍摄，矫正、切割、制作了大量的数据集进行模型训练，在识别屏摄后的字体上取得了很高的训练精度。在准确识别字体的基础上，根据测试得到的误码率选择了合适的纠错编码方案，达到了准确恢复水印信息的效果。为了演示字体识别的准确性，本作品设定了一个易于处理的文档布局方式，并基于该布局方式实现了文档区域识别和单个文字分割，字体识别准确率达到了 92.5%，结合 BCH 纠错码可以准确恢复水印信息。

我们核心的特色有以下四点：

1. 字形的选择：

   通过问卷调查，我们筛选出了人眼难以区分的两种字体用于信息编码。一方面，使用难以区分的字体能够使得水印对文档的视觉效果的影响尽可

能小；另一方面，难以区分的字体可以用于测试字体识别算法的识别能力，从而验证其有效性。

2. 数据集的制作：

   文档在经过手机拍摄后不可避免的会出现各种噪点、模糊的情况。以及因为拍摄距离和拍摄角度等影响，拍摄过后的文档文字上还会出现摩尔纹或产生畸变。我们设计并且实现了畸变矫正算法，并且在采集训练数据的过程中制造了各种环境情况，保证了神经网络的泛化能力足够抵抗大多数的屏摄情况。

3. 深度神经网络的设计：

   采用人眼很难分辨的字体编码信息，需要网络有足够学习能力提取出不同字体的特征。我们设计并且不断改善网络的结构，针对于大量的数据集不断训练增强模型的分类能力。图片的背景颜色和环境因素都会对神经网络的识别特征造成干扰，我们设计并实现了减少背景颜色和噪点等因素影响的数据预处理方案。

4. 水印编码方法的设计：

   文档水印需要保障信息的完整性，所以必须选择有很强的纠错功能的编码方案。我们使用 BCH 编码方案对水印信息进行编码，从而保证即使在字体识别出现错误的情况下仍然能够准确提取水印信息。

5. 微信客户端的实现：

   为了演示这一系统的功能，我们实现了一个微信客户端小程序，用户通过该小程序提交文档、查看水印信息。

第四节　应用前景分析

第一小节　文档溯源

在政府及企业部门中，文档的保密性始终是一个重要的问题，机密文档的泄露将威胁企业及国家的安全，带来极大的风险。

传统的防止文档泄密的方案主要是从源头避免文档的泄露，对于重要文件的打印，可以采用限制打印、复印次数，或指定专人负责，以进行保密，打印文档时需要经过批准，审批，登记等步骤，并定期对纸质文档进行销毁或归档处理。对于重要的电子文档往往采用访问控制、身份认证等手段进行保护。

尽管有这些保密措施，但在用户获得文档的副本后，其去向就难以控制，信息泄露的风险加大。当文档泄露后，如果能定位文档泄露源头并对其进行管控、跟踪和追责，就可以及时避免更大的损失。但随着智能手机的普及，传统的方案已经不能解决屏摄泄密的问题。因此，对文档进行可溯源的处理有着迫切的需求与极大的价值。

要使文档可溯源，就要向其中加入水印。但是随着智能手机的普及，手机拍摄成为一种高危险的泄密方式。原始文档在屏摄处理后会造成大量失真，因此传统的水印信息无法保留。抗屏摄的文档水印技术可以在屏摄过后的图片中恢复出原始的水印信息，文件泄露的图片被捕获后，通过水印信息即可对文档泄密源头进行管控追踪。因此研究这项技术有着极大的价值。

第二小节　信息推送

在商业宣传和海报制作中，设计者往往会在其作品上添加二维码来方便用户获取详细信息，但是二维码形态相对固定，对作品的整体效果有一定的影响，因此在实际使用时人们往往会为其加上多样的形态进行美化。如果能够在艺术宣传作品的文字中直接嵌入网站链接等信息，就能够免去二维码，增加信息维度，从而使设计更加美观而富有科技感。

第二章　作品设计与实现

第一节　系统设计

在核心算法的基础上，我们设计和实现了一个简单的原型系统用以展示我们的文档水印技术。

第一小节　使用流程

1　文档发布

当用户需要发布一份带水印的文档时，首先将文档原文和水印信息提交至本系统，由本系统生成带有水印的文档，然后用户获取该文档，进行后续的处理和使用 (如展示在屏幕上)。

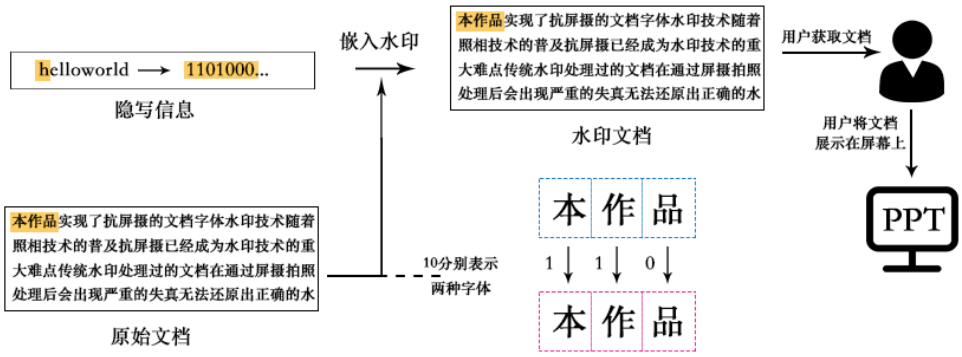


图 1: 文档发布流程图

2　信息提取

当用户需要从带有水印的文档中恢复信息时，首先对文档进行拍照或扫描以得到电子版本，然后将该电子版本提交至系统，系统分析出水印内容后反馈呈现给用户。

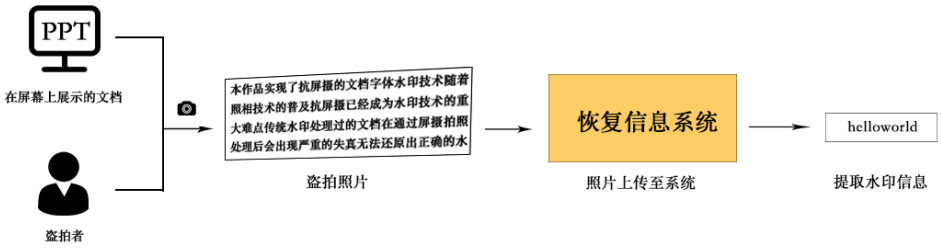


图 2: 水印提取流程图。

第二小节　系统架构

我们的原型系统使用服务端-客户端架构。如下图所示：

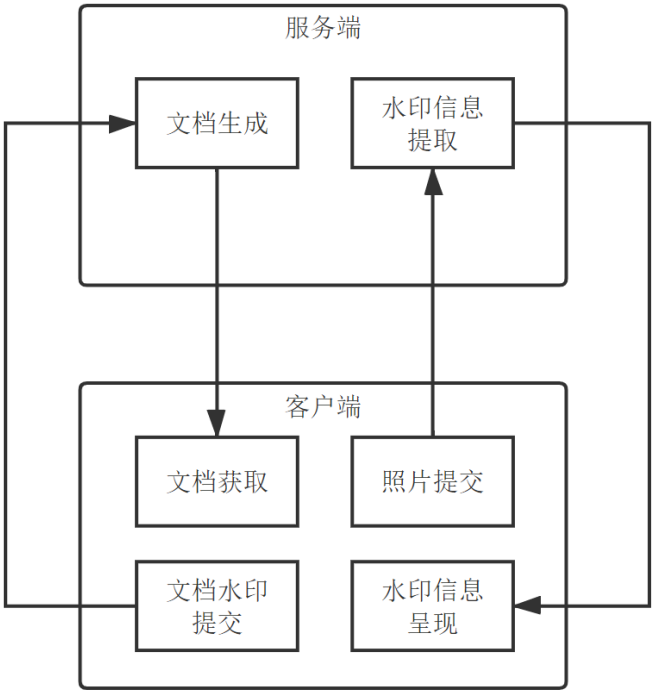


图 3: 系统架构图。

服务端　文档水印和提取的核心算法部署在服务端，此外，服务端含有图片文档切割、矫正、字符切分算法以及与客户端通信的功能。

客户端　向用户提供方便地与系统进行交互的接口，用户通过客户端获取文档、从文档中读取水印信息。客户端以多种形式呈现，包括 Web 页面、微信小程序、手机 App 等，本作品的演示系统中以微信小程序的形式提供客户端界面。

第二节　功能模块

第一小节　文档生成模块

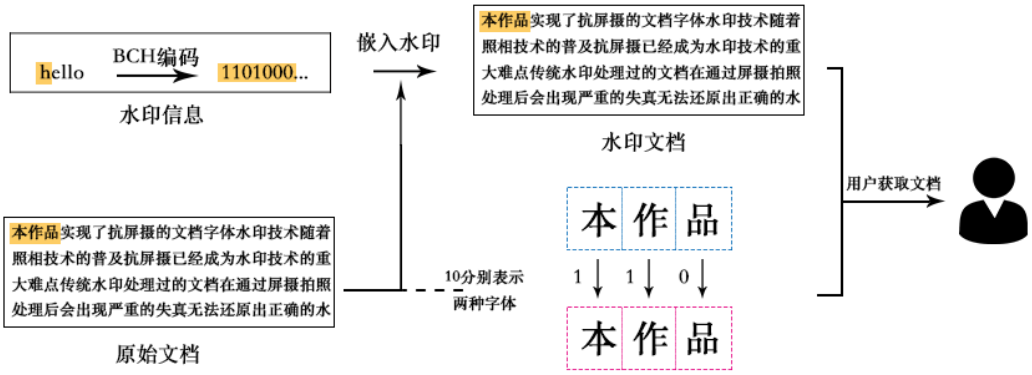


图 4: 文档生成流程。

用户选择需要嵌入的信息 (url) 和嵌入的文档内容。以二进制为例，模块对信息进行 ‘0/1’编码, 将得到的 ‘0/1’比特串进行 BCH 纠错编码得到最终的比特串。文档生成器通过串中的 0 和 1 分别选择两种字体，按照顺序来生成符合内容要求的文档。

第二小节　信息恢复模块

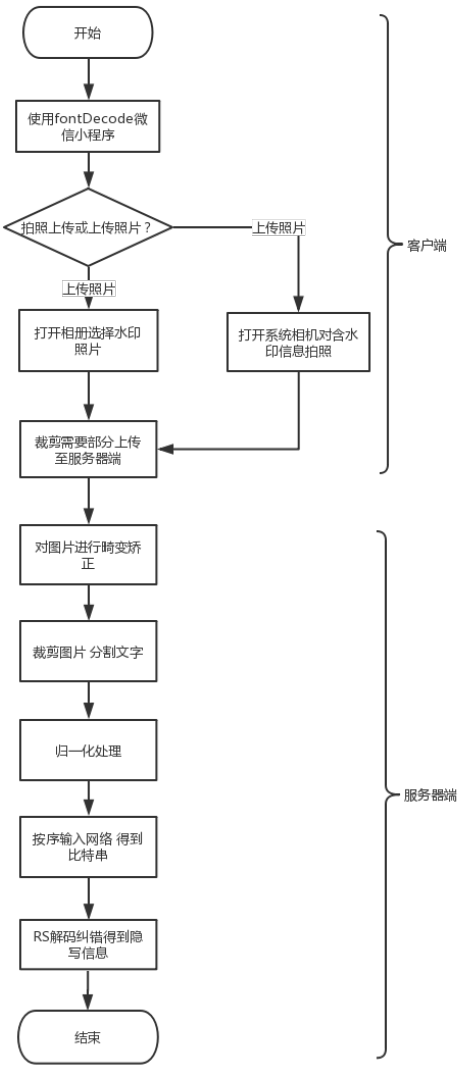


图 5: 信息恢复流程。

信息恢复模块由客户端和服务器端两部分组成。客户端负责上传含有水印的文档的图片。服务器端负责将客户端上传的图片进行处理，最终将解码所得的隐藏信息返回给客户端。

客户端是一个微信小程序，它能允许用户选择相机拍摄上传或选择相册中图片上传。

服务器端对照片进行畸变校正等预处理，形成我们需要的字体排列形式。然后对矫正过后的图片按照设定的文档规格进行切分，得到一张张单个文字图

片。最终，将这些单个字体文件按照顺序输入神经网络分类器中进行分类，按照字体类别得到‘0/1’比特串。将比特串输入 BCH 解码器解码后，恢复原有信息。

第三节　技术实现

第一小节　文档生成算法

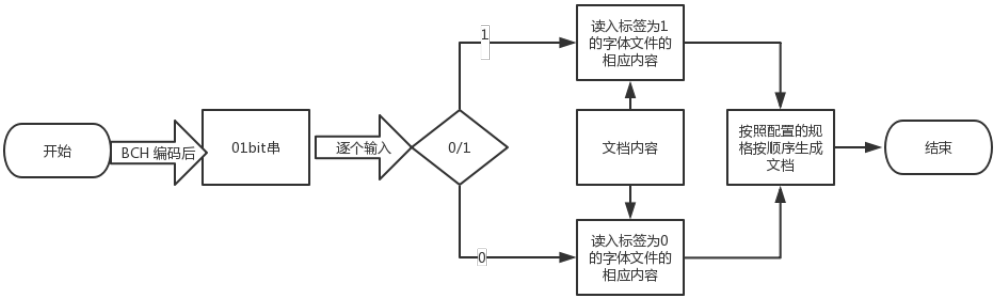


图 6: 文档生成算法流程

该算法的核心是一个信息获取器和文档生成器。根据输入的比特是 0 还是 1，来选择生成文档中对应文字的字体。‘0\1’比特串输入结束后，文档生成器将结果显示出来。

第二小节　图像矫正算法

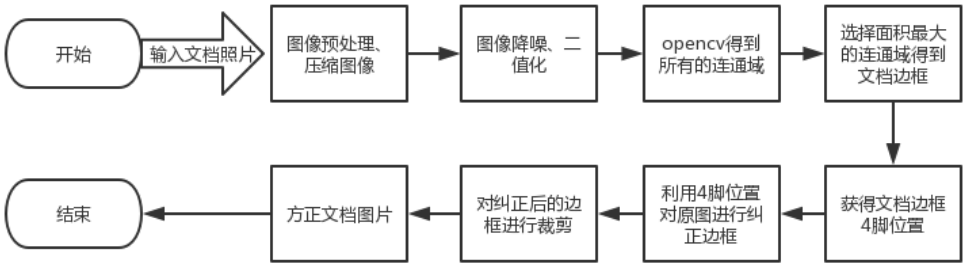


图 7: 图片矫正算法。

该算法将输入的照片依次进行如下操作：图片灰度化、阈值二值化、检测轮廓、寻找轮廓的包围矩阵。获取角度，根据角度进行旋转矫正。随后对旋转后的图像进行轮廓提取，裁剪出轮廓内的图像区域，成为一张独立图像。本算法采用连通域识别的方法：识别文档照片的边框，并根据边框的位置进行几何矫正，最后得到矫正后的图片。

1 连通域识别

在图像中，最小的单位是像素，每个像素周围有 8 个邻接像素，常见的邻接关系有 2 种：4 邻接与 8 邻接。4 邻接一共 4 个点，即上下左右，如下左图所示。8 邻接的点一共有 8 个，包括了对角线位置的点，如下右图所示。

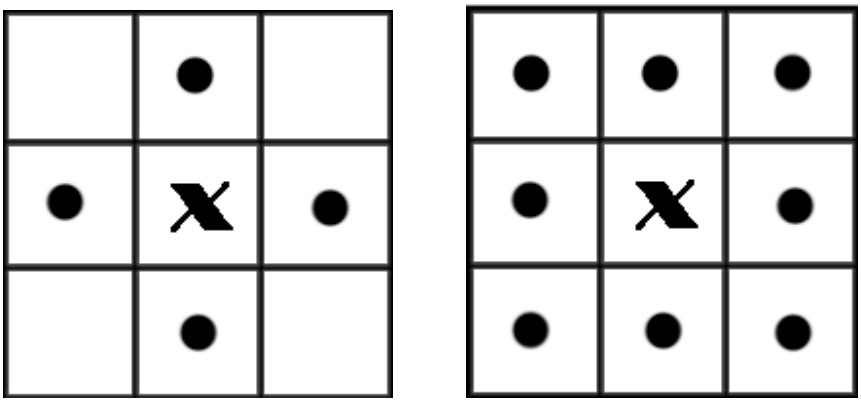


图 8: 连通域示例。

如果像素点 A 与 B 邻接，我们称 A 与 B 连通，于是我们有如下的结论：如果 A 与 B 连通，B 与 C 连通，则 A 与 C 连通。在视觉上看来，彼此连通的点形成了一个区域，而不连通的点形成了不同的区域。这样的一个所有的点彼此连通点构成的集合，我们称为一个连通区域。下面这符图中，如果考虑 4 邻接，则有 3 个连通区域；如果考虑 8 邻接，则有 2 个连通区域。（注：图像是被放大的效果，图像正方形实际只有 4 个像素）。

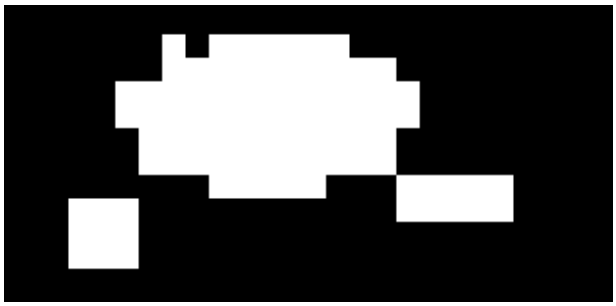


图 9: 连通域示例

连通域识别和提取算法中使用了图像形式化处理的方法，介绍其中的几个概念：

膨胀：

作为 Z^2 中的集合 A 和 B，

中的集合 A 和 B，表示为的 B 对 A 的膨胀定义为

$$\mathrm{A} \oplus \mathrm{B} = \left\{ \mathrm{z} | (\hat{B})_z \cap \mathrm{A} = \emptyset \right\} \tag{1}$$

如图所示：

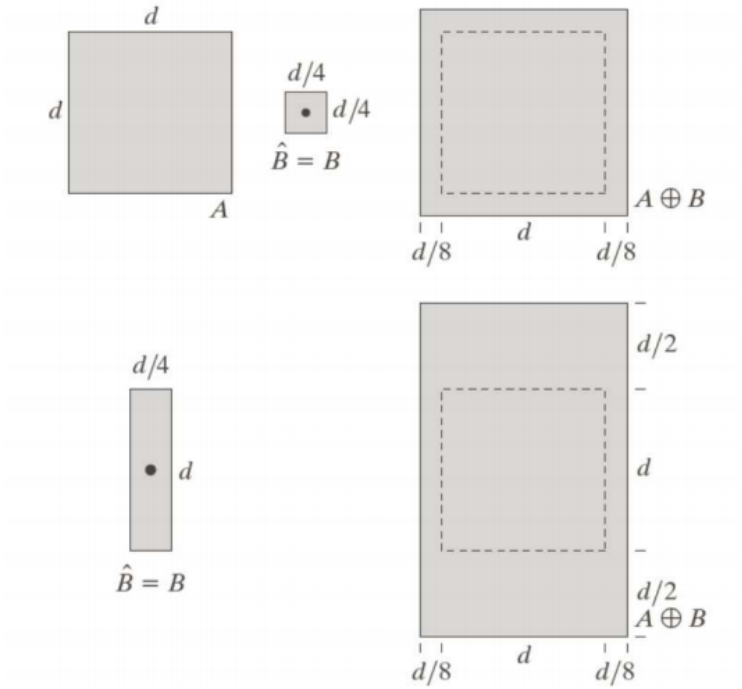


图 10: 形式化处理中的膨胀过程

同理膨胀过程可以表示为：

$$A \oplus B = \left\{ z | \left[(\hat{B})_z \cap A \right] \subseteq A \right\} \tag{2}$$

有了膨胀的概念，则连通分量的提取算法如下所示：

算法 1 连通域提取算法

1. 对图像进行二值化处理
2. 除了在对应于 A 中每个连通分量的一个点的每个已知位置处我们已置为 1 外，该阵列的所有其他元素均为 0
3. 迭代：

$$X_k = (X_{k-1} \oplus \mathrm{B} \cap \mathrm{A}) \quad k = 1, 2, 3, \cdots \tag{3}$$

4. 直到 $X_k = X_{k-1}$ 时停止迭代
5. X_k 包含输入图像中的所有连通分量

下图可以完整的体现这个过程：

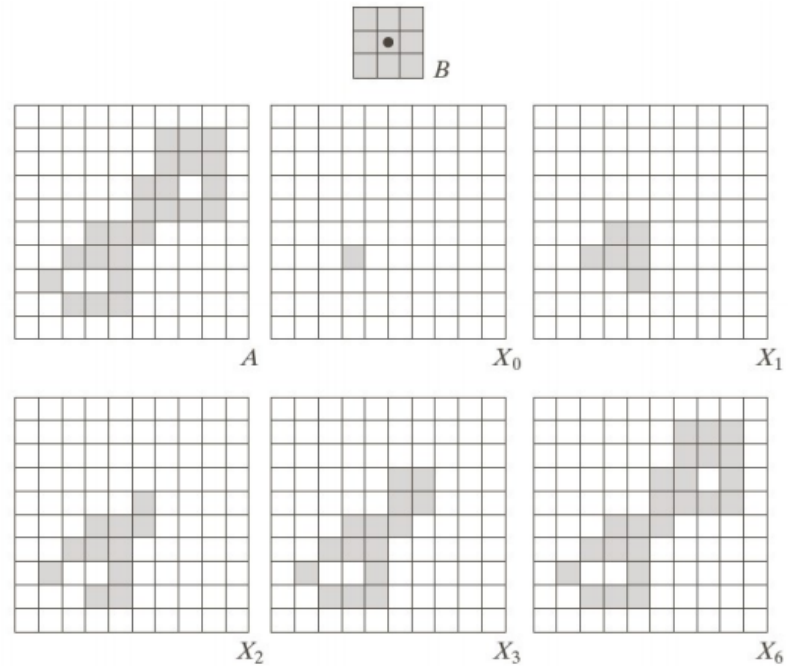


图 11: 连通域提取过程

2 几何校正

仿射变换和透视变换更直观的叫法可以叫做「平面变换」和「空间变换」或者「二维坐标变换」和「三维坐标变换」，一个是二维坐标（x,y），一个是三维坐标（x,y,z）。也就是：

1. 仿射变换：

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{4}$$

$$\begin{aligned} x &= a_{11}u + a_{12}v + b_1 \\ y &= a_{21}u + a_{22}v + b_2 \end{aligned} \tag{5}$$

2. 透视变换：

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{6}$$

$$\begin{aligned} x &= a_{11}u + a_{12}v + a_{13} \\ y &= a_{21}u + a_{22}v + a_{23} \\ z &= a_{31}u + a_{32}v + a_{33} \end{aligned} \tag{7}$$

$$\begin{aligned} x &= a_{11}u + a_{12}v + a_{13} \\ y &= a_{21}u + a_{22}v + a_{23} \\ z &= a_{31}u + a_{32}v + a_{33} \end{aligned} \tag{8}$$

$$x' = \frac{x}{z} = \frac{a_{11}u + a_{12}v + a_{13}}{a_{31}u + a_{32}v + a_{33}} = \frac{k_{11}u + k_{12}v + k_{13}}{k_{31} + k_{32}v + 1} \tag{9}$$

$$y' = \frac{y}{z} = \frac{a_{21}u + a_{22}v + a_{23}}{a_{31}u + a_{32}v + a_{33}} = \frac{k_{21}u + k_{22}v + k_{23}}{k_{31} + k_{32}v + 1} \tag{10}$$

摄像机就相当于透视中心，摄像机和荧幕之间就是像点的集合，荧幕上展示的就是目标点集合；中心点和某一像点的连线和荧幕的交点就是这个像点的目标点；像点 (x, y) 和目标点 (s, t) 是一一对应的关系，我们可以使用一个 [3x3] 的矩阵 A 描述他们之间的映射关系：$[s, t, 1]T = dot(A, [x, y, 1]T)$ --- A 矩阵乘 (x, y) 的齐次坐标的转置就是 (s, t) 的齐次坐标的转置，每一个透视变换都对应着特定的变换矩阵 A。

第三小节　字符切割算法

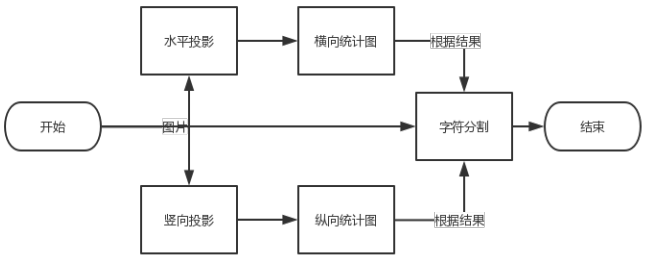


图 12: 字符切分流程

算法如下：

算法 2 字符切分算法

1. 对图片进行水平投影，找到每一行的上界限和下界限，进行行切割
2. 对切割出来的每一行，进行垂直投影。
3. 找到每一个字符的左右边界，进行单个字符的切割

水平投影和竖向投影的结果如图所示：

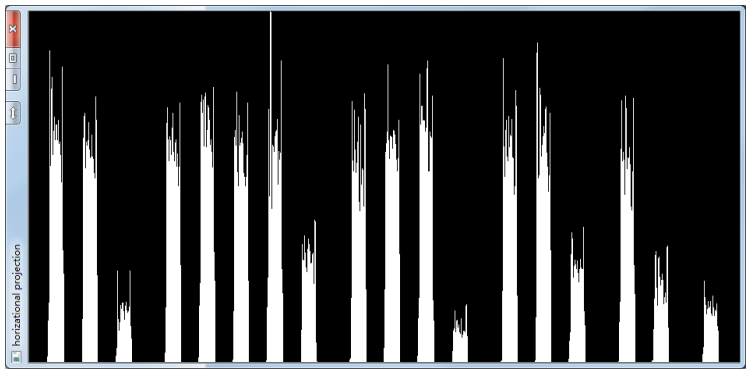


图 13: 投影结果

第四小节　BCH 编解码算法

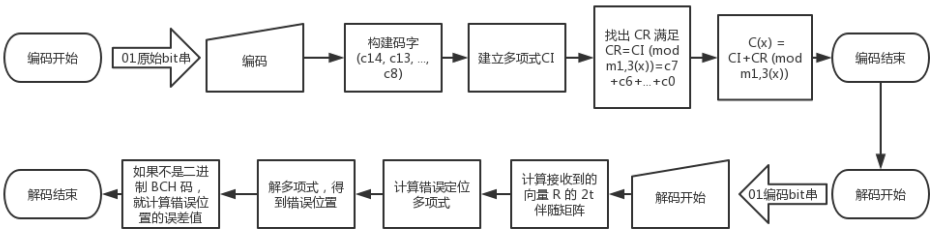


图 14: BCH 编解码算法。

1　构建

BCH 码使用有限域上的域论与多项式。为了检测错误可以构建一个检测多项式，这样接收端就可以检测是否有错误发生。要构建一个能够检测、校正两个错误的 BCH 码，我们要使用有限域 $GF(16)$ 或者 $Z^2[x]/(x^4+x+1)$。如果 α 是 $m_1(x)=x^4+x+1$ 的一个根，那么 m_1 就是 α 的极小多项式，这是因为 $m_1(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)=x^4+x+1$。如果要构建一

个能够纠正一个错误的 BCH 码，那么就使用 $m_1(x)$，这个代码就是所有满足 $C(x) \equiv 0\ mod\ m_1(x)$ 且根为 $\alpha, \alpha^2, \alpha^4, \alpha^8$ 的多项式 $C(x)$。

2 解码算法

解码算法如下所示

1. 首先生成 $2t$ 伴随矩阵然后生成元素为 $S_{t\times t} = \begin{bmatrix} s_1 & s_2 & s_3 & \dots & s_t \\ s_2 & s_3 & s_4\dots & \dots & s_{t+1} \\ s_3 & s_4 & s_5 & \dots & s_{t+2} \\ \dots & \dots & \dots & \dots & \dots \\ s_t & s_{t+1} & s_{t+2} & \dots & s_{2t-1} \end{bmatrix}$ 的矩阵S_{txt}

2. 生成元素为 $C_{t\times 1} = \begin{bmatrix} s_{t+1} \\ s_{t+2} \\ \dots \\ \dots \\ s_{2t} \end{bmatrix}$ 的矩阵 c_{tx1}

3. 让 Λ表示未知的多项式系数，用$\Lambda_{t\times 1} = \begin{bmatrix} \lambda_t \\ \lambda_{t-1} \\ \dots \\ \lambda_3 \\ \lambda_2 \\ \lambda_1 \end{bmatrix}$ 表示

4. 这样就得到矩阵方程 $S_{t\times t}\Lambda_{t\times 1} = C_{t\times 1}$

5. 如果矩阵 $S_{t\times t}$ 存在行列式，那么我们就可以找到这个矩阵的逆，然后就可以得到 Λ 的值如果 $det(S_{t\times t}) = 0$，那么按照以下运算法则进行计算

---

if $t = 0$
then declare an empty error locator polynomial
stop Peterson procedure.
end
set $t \leftarrow t - 1$
continue from the beginning of Peterson's decoding

---

6. 在 Λ 的值确定之后，自然就得到错误定位多项式

7. 结束

第五小节　网络模型训练

1　网络结构

网络中主要结构来自于 ResNet18[8]。网络结构中有很多 block 组成，每个 block 的构成如下图所示：

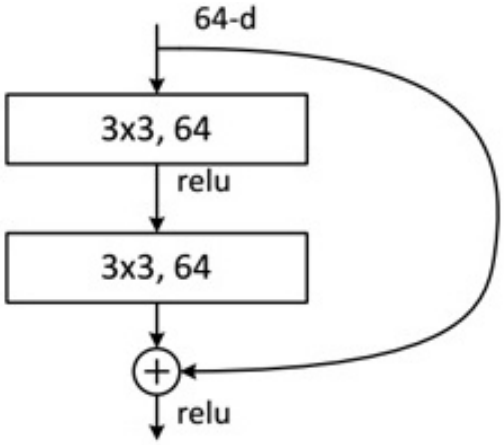


图 15: 残差单元。

网络使用残差单元作为 block。假设设输入为 x，传统网络单元的输出为 $H(x)$，那么残差单元的输出为：

$$F(x) = H(x) - x, \tag{11}$$

引入残差后的映射对输出的变化更敏感，所以效果更好。残差的思想是去掉相同的主体部分，从而突出微小的变化。网络整体结构如下图所示：

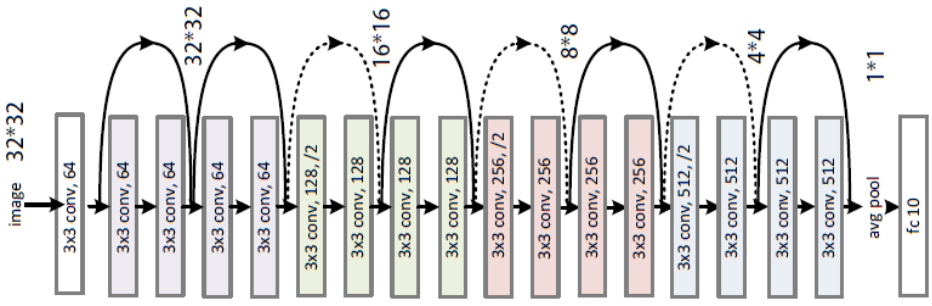


图 16: 神经网络模型。

2 训练数据

神经网络的训练需要足量的数据，否则会出现欠拟合（Underfitting）的问题。目前并没有符合本作品需求的中文字体数据集，因此本小组选择自己制造训练和测试用的数据。由于需要多种字体数据进行训练和测试，下面的步骤针对每种字体重复进行：

生成文档 首先需要批量化地生成“文档”。为了简化与关键技术关系不大的部分，本小组使用了简单的规则生成“文档”来解决下述问题，以方便文档矫正和文字切割，针对更复杂的文档格式、布局的处理不在本次作品的研究范围内。

文档格式设计规则如下：

1. 对于任意指定的文档，从中分割出所有汉字并不容易，因此规定简单文档格式每一页的文字行数、列数固定，且不含非汉字字符，如标点、空格、英文字母等；
2. 文档在屏幕上显示时，边界有时不容易确定，因此在每页文档周边加一个黑色边框；
3. 文档从屏幕或纸张上被拍照后，其页面边界会有一些影响文档边界自动识别的因素，导致边界出现微小偏移，为了降低其对切割出的文字的影响，在文字与文字之间加入一定大小的空白。

下图是一页文档的示例：

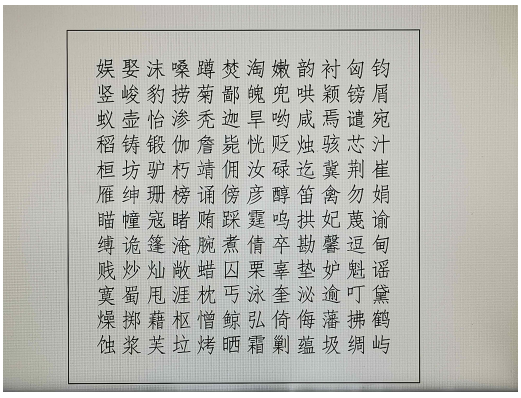


图 17: 屏摄文档示例。

屏幕拍照 考虑到用户在拍照时，光照、拍照角度、显示文档的屏幕等因素都很难确定，在制作训练数据时体现出这样的多样性才能使训练出的网络模型具有较好的泛化能力，因此制作训练数据时我们做了以下的变化：

1. 从多个角度拍照；
2. 变化背景光照、屏幕亮度拍照；
3. 使用不同的屏幕进行拍照；
4. 从不同的距离拍照。

文档矫正　使用传统算法进行边界识别，然后做透视畸变矫正，得到照片中的文档区域，对于“生成文档”中所示图片，矫正后得到下图：

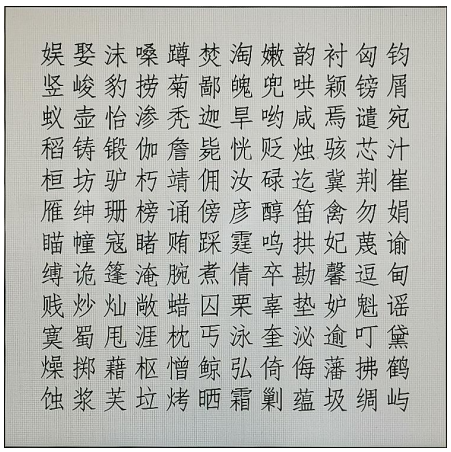
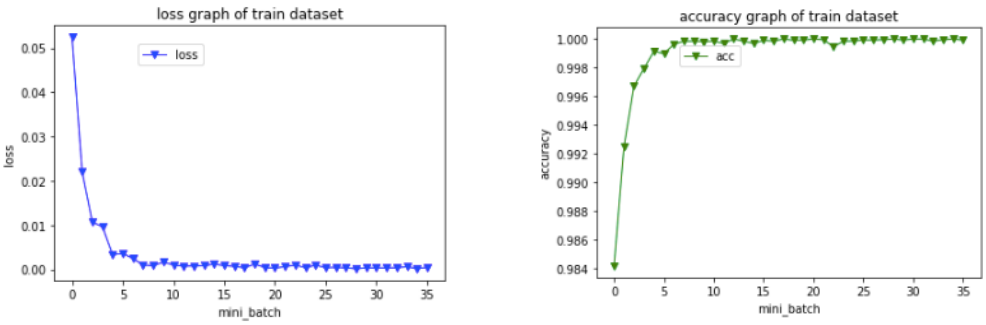


图 18: 矫正后的文档示例。

文字切割　在矫正后的图像上按照预定的行列数进行均等划分，得到一个一个字的图像。

网络训练　将切分好的图像处理为神经网络能接受的格式，随机输入进神经网络，利用 BP 算法进行训练。在参数选择上，我们选择 batch size 为 64，epoch 为 12，学习率为 0.001。我们选用 CrossEntropy 作为损失函数，SGD 作为优化器算法。

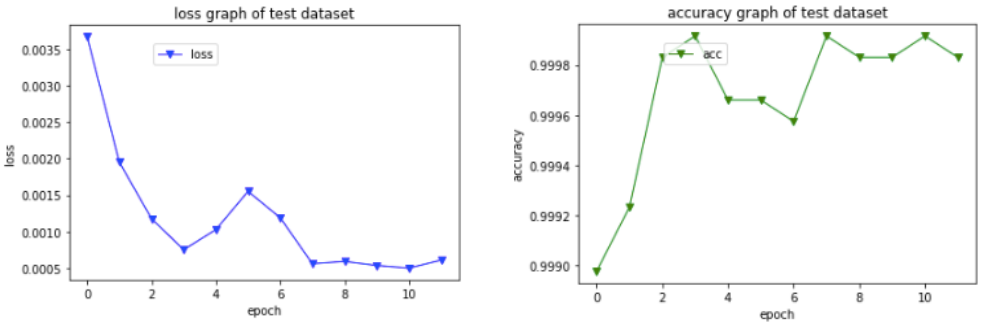
在学习过程中，我们每个 minibatch（400 个 batch）计算一次训练集 loss 和 accuracy。某一次训练的结果如下图所示：



(a) 训练集 loss，　　(b) 训练集 accuracy。

图 19: 训练集结果。

每个 epoch 计算一次测试集 loss 和 accuracy，结果如下图所示：



(a) 测试集 loss，　　(b) 测试集 accuracy。

图 20: 测试集结果。

从图中可看出训练集 loss 和 accuracy 一直在减小和增加。然而测试集 loss 出现了一个增加高峰，accuracy 出现了一个降低的低谷。我们判断在第 3 个 epoch 训练出现了过拟合，于是选择了第 3 次 epoch 训练出模型进行之后的测试与使用。

第四节　系统展示

下面展示某一次系统的使用演示：

获得文档生成

我们采用三中的一段话作为原始文本，‘ciscn!’ 作为待水印信息。将原始文本和给定信息输入文档生成模块，得到的文档如下图所示：

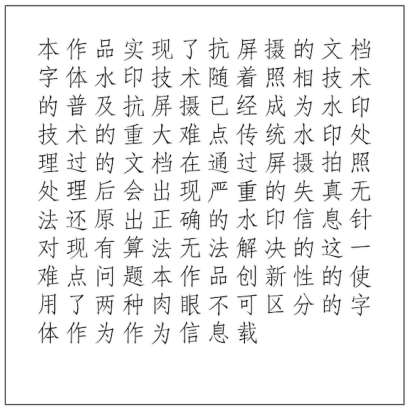
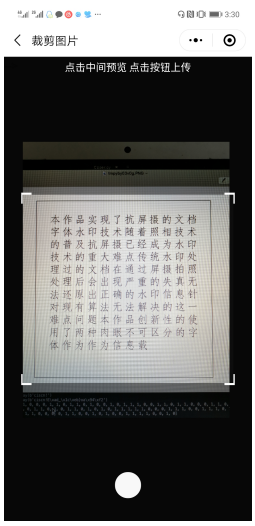


图 21: 带有水印的文档

识别文档照片中的隐藏信息

打开微信搜索 fontDecode 小程序，点击中间按钮，你可以选择上传相册中的照片或者临时拍照上传。

(a) 主页



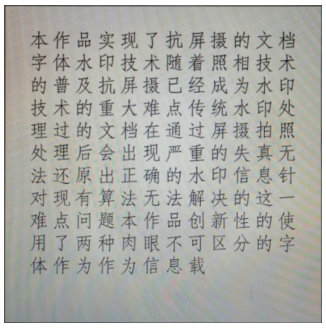
(b) 裁减页

图 22: fontDecode 小程序

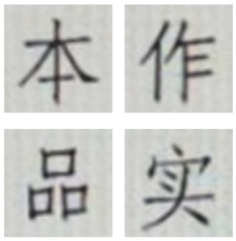
这里我们选择演示拍照上传。打开系统相机后，拍摄屏幕上含有水印的文档（模拟现实拍照场景）。之后，旋转缩放并切割拍到的照片。点击下方按钮将处理后的图片上传至服务器。

服务器收到上传图片后，先进行定位、畸变矫正、切割。将切割后的图片规范到 224 x 224 并进行归一化处理，依次输入进网络。将网络输出的比特串进

行 BCH 解码，即能得到图片中的隐藏信息。



(a) 服务器畸变矫正后的图像



(b) 切割后的图像

图 23: 服务器端处理

之后服务器会将解码得到的信息发回给客户端，用户成功获得隐藏信息。

图 24: 溯源结果

第三章　作品测试与分析

本作品可以使用任意多种类的字体作为信息的载体，在本次实现和测试中，为了突出核心想法和测试信息恢复的准确性，选择了两种非常相近的字体。其中一种为华文仿宋，另一种为微软简仿宋。

第一节　测试环境

第一小节　硬件环境

移动端　考虑到微信小程序已具备各手机平台适应性，我们仅选用 MI 8、Huawei mate20、360 N6、vivoX7 作为移动端测试平台，测试其在微信小程序内打开系统相机、上传图片、裁剪图片和与服务器使用 https 协议通信的可能性。

不同手机自带的摄像头在性能方面也有不同，如像素值、对焦能力等。因此我们在采集数据时，混合了四款手机拍照得到的照片进行测试训练。增加神经网络识别不同手机拍摄的图片的能力。

显示屏　考虑不同显示屏的分辨率、色差等会对图片产生影响，我们在 HP Pavilion 15-au147tx、mac air13.3-inch (1440 x 900)、mac pro13.3-inch (2560 x 1600) 均进行了采样，并在这些显示屏上拍取最终编码后的图片进行解码测试。

服务器端　服务器端配置为 Ubuntu 16.04（64 位、1 核、内存 2G）。服务使用 nginx+gunicorn+flask 进行部署环境。经测试，该环境足够为小量用户提供可靠服务。

第二小节　自然环境

光照环境　光照条件不同对拍照背景颜色有很大影响。在测试网络识别能力的过程中，我们添加了不同光照条件下拍摄的测试图片，测试网络能否去除光照引起的背景颜色不同的影响。

拍照距离　由于距离上的差别，拍照所得到的图像在清晰度、摩尔纹等方面有很大的不同。为了测试我们的字体识别网络是否具有处理摩尔纹和清晰度的能力，我们在拍取测试图片时时随机选取拍照距离。需要注意的是，选取的距离不应过大，使得图片中文字无法识别。也不应过小，使得对整张图片截取不完全，从而无法解码。

旋转倾斜角度　为模拟现实情况下拍照环境，我们在拍照时将手机随机进行了小幅度旋转倾斜，以此测试文档分割前图片畸变矫正算法的鲁棒性。

抖动　测试数据中包含手持拍摄和三脚架固定拍摄数据，以检测网络对模糊图像的识别率。需要注意的是，抖动程度不应太大导致图片文字模糊不清。

第二节　测试方案与结果

分单元分别对字体类型的不可见性、屏摄图片分割准确性和水印信息嵌入与提取进行测试，并对整体流程进行完整测试与分析。

第一小节　功能测试

我们模拟了数次完整的使用流程，包括文档生成模块测试和信息恢复模块测试。先测试使用给定原始文本和水印内容，使用文档生成模块产生带有水印的文档。再对该文档使用客户端拍照上传，服务器端进行定位、畸变矫正、字符分割处理后，将字按序输入网络，将输出的比特穿进行 BCH 解码，就能得到水印信息。下面展示某一次的详细使用流程：

1　文档生成模块测试

我们随机生成 20 个 6 字符长的英文字符串。为这 20 个字符串分别生成不同个数的文档，各文档均能正常生成。各字符串以及对应的文档个数如下：

y)PɑE*	4@P\Om	&Xo$\R	A[F~Rr	p’>wSf
3	2	4	5	3
gIx%c>	w^:&‘X	OMqIGc	!\>hrI	S5_h|[
2	4	5	3	2
bnuRXx	T}s\;$	KZ!#v5	S6Uq7&	MNZY%S
4	5	3	2	4
"f:<t@	}pOUxu	g:ɑZ!w	7\(Kp5	o_6]%(
5	3	2	4	5

表 1: 水印信息及对应文档数

2 信息恢复模块测试

我们的 fontDecode 小程序中加入了 4 个试用者，分别是 A、B、C、D。让 4 位试用者使用 MI 8、Huawei mate20、360 N6、vivoX7 对之前产生的 70 张图片拍照，上传至服务器提取信息。使用过程和系统展示四中相同。结果如下：

	A（Huawei mate20）	B（MI 8）	C（360 N6）	D（vivo X7）
成功恢复文档数	67	65	64	60

表 2: 信息恢复效果

我们推测还原数目不同与拍摄时抖动程度与手机像素值有关。

第二小节　单元测试

1 字体人眼不可识别性测试

测试方法及标准　作为文档水印，两种编码字体不能被人眼区分是非常重要的。为了确定我们选定字体的差异是否是难以察觉的，我们使用了主观评测的形式。若两种字体的平均相像评分到达特定阈值，我们便可以认为选择的两种字体是可以用来编码的。

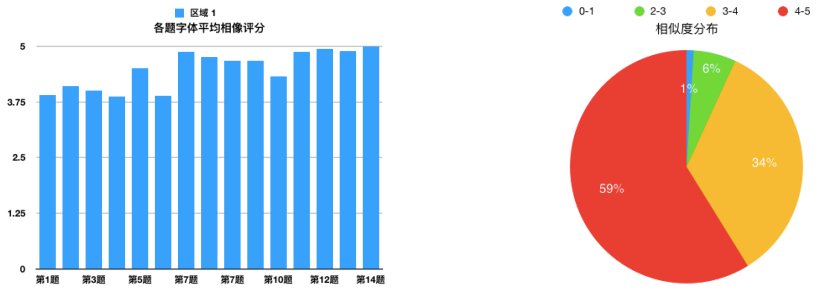
测试细节　我们选定 4 分作为阈值，并随机抽取两个相同文字让问卷参与者进行辨别（如左图所示）。并且，为了防止参与者中有恶意用户，我们设置了四组验证题（如右图所示）。正常问题中的两个文字分别为我们选定的两种字体，验证问题中的两个文字则为相同字体。若参与者在四个验证问题中有两个问题回答的评分低于 3.5 分，我们则认为他是恶意用户，该用户的回答不计入结果计算。

(a) 正常问题　　(b) 验证问题

图 25: 问题示例。

测试结果　我们对每个问题的结果（前十题为正常题，后四题为验证题）和去除恶意用户后的总体结果分别进行了统计，统计结果如下图所示。总体上，我们选定的两种字体的平均相像评分达到了 4.097，超过了设定的阈值。

实际使用过程中，几乎不可能出现两个不同字体的相同文字相邻出现的情况。因此我们的字体不可区分性将表现的更为出色。



(a) 各题平均分数　　(b) 各分数段分布

图 26: 问卷调查结果

2　字体识别准确率测试

测试方法及标准　准备一组屏摄切割后的文字，识别其字体，计算准确率。准确率计算公式如下：

$$M = \frac{P_1 + P_2}{T_1 + T_2}, \tag{12}$$

其中 P_1、P_2 为两种字体分别正确识别的数目，T_1、T_2 为两种字体图片的数目。

屏摄切割后的文字示例如下：

图 27: 屏摄切割后的文字图像示例

测试细节　我们随机选择了 12000 张字体图片用做测试。最终的字体识别出错率应低于 BCH(127, 50) 的 BER——10%，即字体准确率应高于 90%。

测试结果　12000 张字体图片的识别准确率为 92%，在准确率允许范围内。

3　文档分割矫正正确性测试

测试方法及标准　准备一组（超过 300 张）屏摄文档的照片，用我们的文档分割算法进行分割，计算分割的准确率。

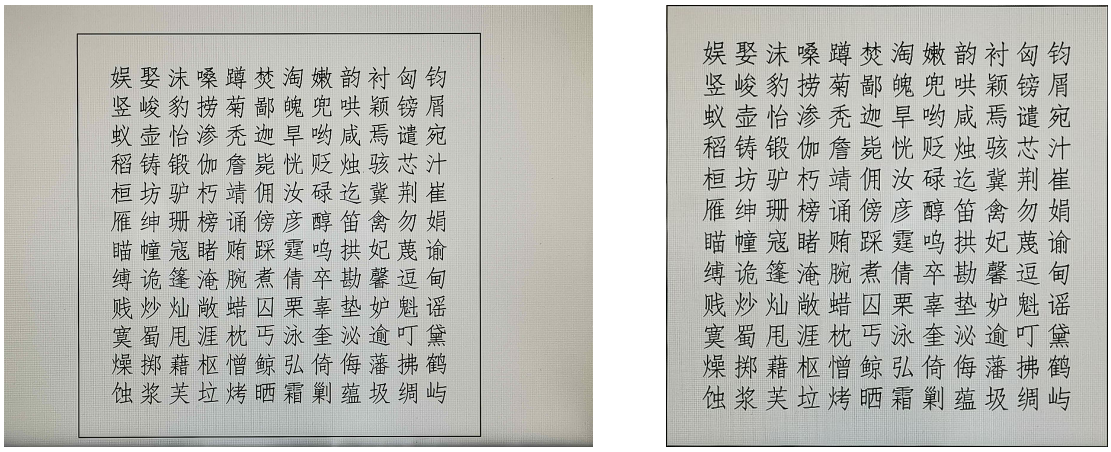


图 28: 分割前后图像示例

测试细节　我们对不同旋转角度的图片进行测试，也对不同距离进行了拍照测试。

测试结果　旋转角度小于等于正负 10 度、不遮挡边框的情况下，文档均能被正常分割矫正。

4　文档解码正确率测试

测试方法及标准　随机生成数据（‘0/1’比特串），并对数据进行 BCH 编码。利用编码后的比特串产生隐写文档。对文档进行拍照、畸变矫正、切分后，使用2中达到要求的网络提取出信息，将提取信息与文档中真实隐写信息比对，计算识别准确率。最终将提取信息进行 BCH 解码，观察解码后的数据与一开始随机生成的数据是否相同。

测试细节　提取信息与文档中真实隐写信息比对的准确率应高于 90% 才能正确解码。我们随机产生了三组数据（‘0/1’比特串），对每组数据产生了 12 份隐写文档，每份文档中含有 144 个文字。

测试结果　结果如下表所示：

表 3: 文档解码正确率结果

	数据 1	数据 2	数据 3
平均识别率	93.2%	95%	92.1%
正确解码的文档数	11	12	11

所有文档中只有两份文档识别正确率为 88%，无法被正确 BCH 解码。我们的系统正确提取隐写信息的概率为 93.75%

第四章　创新性说明

本作品创新性地将字体作为信息的载体，利用深度学习技术灵活强大的模式识别能力对不同的字体进行精准识别，结合纠错编码，保证了水印信息的无损恢复。从而实现了一种能抵抗屏摄过程的文档水印技术。

本技术具有如下特点：

1. 强可定制性

   本作品提出的技术可适用于任意字体类型、任意字体数目，因此具有很高的可定制性，可以适应于多种文档格式，文字类型等需求，具有广泛的应用价值。

2. 高不可见性

   实验表明，本作品提出的技术在肉眼难以区分的前提下，字体识别准确率也能够超过 90%，配合纠错编码，可以无损恢复水印信息。因此使用该技术不会对文档本身造成肉眼可见的失真，也不会影响文档的正常阅读和使用。

3. 高鲁棒性

   在纸质、电子文档的传播过程中可能经过打印、扫描、屏幕显示、拍照等多种变化过程，在这些过程中很容易在文档中留下噪点、污点。传统的分析方法集中于在文档中使用像素点记录信息，很容易受到噪点、污点的影响，导致信息丢失、出错。本作品使用字体作为信息的载体，有效避免了信息被污点、噪点干扰的问题。

   本作品从两个方面提高了水印提取的准确性。一方面，使用字体作为信息载体，文档的水印信息不再容易受到污点、噪点、甚至屏幕拍照出现的摩尔纹的影响；另一方面，深度学习技术使得字体识别更为灵活准确。

4. 大信息容量

   虽然在实现和测试时我们只使用了两种字体，但实际上，只要不影响文档阅读效果，可以使用更多字体，从而增加每个文字信息载荷，进而提高同样长度的文档对应的水印容量。

第五章　总结

多媒体技术的发展对泄密溯源技术带来新的挑战。随着智能手机的普及，各类重要的机密文件的泄露事件出现频繁。目前各种文件几乎都以电子文档的形式存储，文档显示在电子屏幕上时，信息窃取者通过手机拍摄照片获取文档的信息变得十分容易，而这种文档泄密事件的溯源却相当困难。屏摄后的文档图片会出现大量的摩尔纹、噪点、失真等情况，传统的文档水印技术无法在屏摄后恢复原始的水印信息。因此需要一种能够抗屏摄的文档水印技术。

我们提出了一种具有高不可，性、高鲁棒性且抗屏摄的中文文档水印方案。并实现了一个原型系统来演示其效果。我们通过众包技术选取了人眼不可区分的字体作为编码信息的载体，通过具有纠错能力的编码方案 BCH(127,50) 将文档中嵌入水印信息。水印提取过程中，使用畸变矫正算法来处理屏摄后的图片，利用 OCR 技术裁剪获取文字。利用神经网络识别字体种类，将提取出的信息利用 BCH(127,50) 解码还原出原始的水印信息。通过在神经网络的训练过程中，我们拍摄了大量的不同环境的训练集来提高网络的泛化能力，最后通过单元测试验证了网络的高识别准确率和提取信息的正确性。演示系统基于屏摄文档，但可以扩展到其他形式的文档，具有普遍应用的潜力。

未来我们的抗屏摄文档水印技术还可以利用生成对抗网络在字体选择这一部分进行优化：现在我们的原型系统仅支持特定的字体。当用户任意给定字体时，使用众包技术为其选择相像字体显然是不现实的（周期太长、无法支持大规模使用）。针对这个问题，我们可以使用生成对抗模型，对任意给定的字体都生成一种与其相似的、我们的网络可以识别的字体。在文档篇幅较小的情况使用‘0/1’编码无法嵌入足够的水印信息，对于这个问题，可以利用对抗生成网络产生大量的相似字体，从而提高每一个可以编码的信息，依然可以实现水印信息的嵌入。

参考文献

[1] Chang Xiao, Cheng Zhang, and Changxi Zheng. Fontcode: Embedding information in text documents using glyph perturbation. ACM Transactions on Graphics (TOG), 37(2):15, 2018.

[2] Peter O'Donovan, Jānis Lībeks, Aseem Agarwala, and Aaron Hertzmann. Exploratory font selection using crowdsourced attributes. ACM Transactions on Graphics (TOG), 33(4):92, 2014.

[3] Jörn Loviscach. The universe of fonts, charted by machine. In ACM SIGGRAPH 2010 Talks, page 27. ACM, 2010.

[4] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In Proceedings of the European Conference on Computer Vision (ECCV), pages 657 – 672, 2018.

[5] Carlos Avilés-Cruz, Risto Rangel-Kuoppa, Mario Reyes-Ayala, A Andrade-Gonzalez, and Rafael Escarela-Perez. High-order statistical texture analysis—-font recognition applied. Pattern Recognition Letters, 26(2):135 – 145, 2005.

[6] Guang Chen, Jianchao Yang, Hailin Jin, Jonathan Brandt, Eli Shechtman, Aseem Agarwala, and Tony X Han. Large-scale visual font recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3598 – 3605, 2014.

[7] Zhangyang Wang, Jianchao Yang, Hailin Jin, Jonathan Brandt, Eli Shechtman, Aseem Agarwala, Zhaowen Wang, Yuyan Song, Joseph Hsieh, Sarah Kong, et al. Deepfont: A system for font recognition and similarity. In 23rd ACM International Conference on Multimedia, MM 2015, pages 813 – 814. Association for Computing Machinery, Inc, 2015.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770 – 778, 2016.

[9] Peter Wayner. Mimic functions. Cryptologia, 16(3):193 – 214, 1992.

[10] Peter Wayner. Disappearing cryptography: information hiding: steganography and watermarking. Morgan Kaufmann, 2009.

[11] 黄兴. 抗打印扫描的文本数字水印技术研究. Master's thesis, 北京邮电大学, 2013.

[12] Jeebananda Panda, Nishant Gupta, Parag Saxena, Shubham Agrawal, Surabhi Jain, and Asok Bhattacharyya. Text watermarking using sinusoidal greyscale variations of font based on alphabet count. International Journal of Innovative Research in Computer and Communication Engineering, 3(4):3353 – 3361, 2015.

[13] Raj Chandra Bose and Dwijendra K Ray-Chaudhuri. On a class of error correcting binary group codes. Information and control, 3(1):68 – 79, 1960.