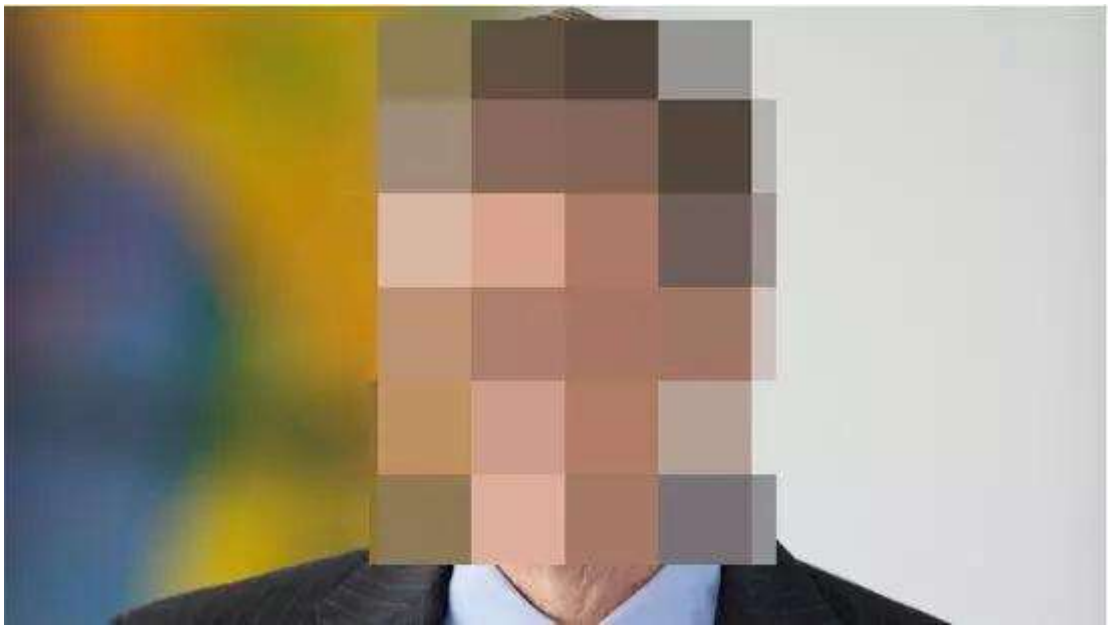
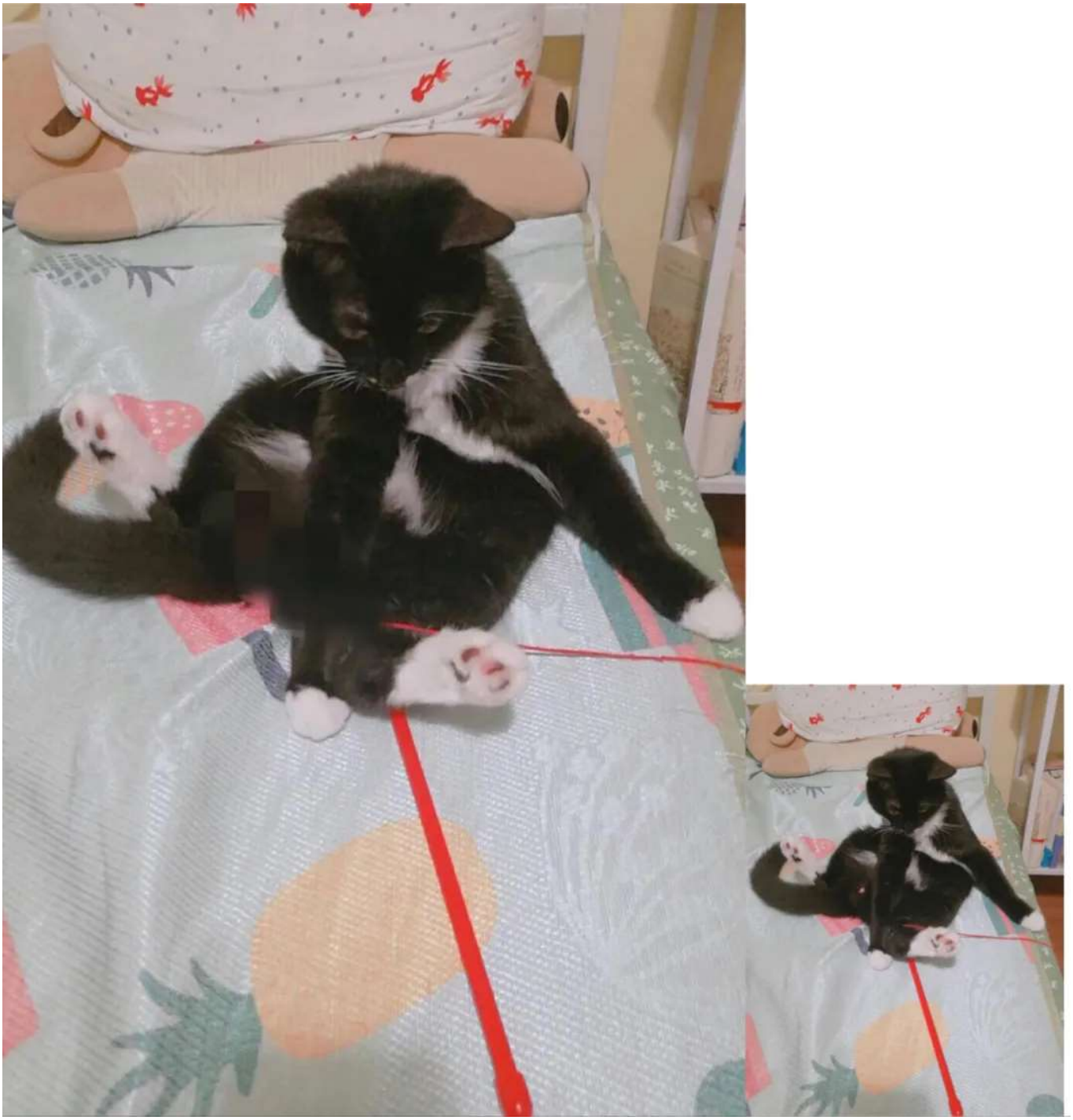
> 还在用马赛克的方式隐藏密码？小心被「看穿」。

像素化（又称马赛克）是一种常见的打码方式，通过降低图像中部分区域的分辨率来隐藏某些关键信息，比如：



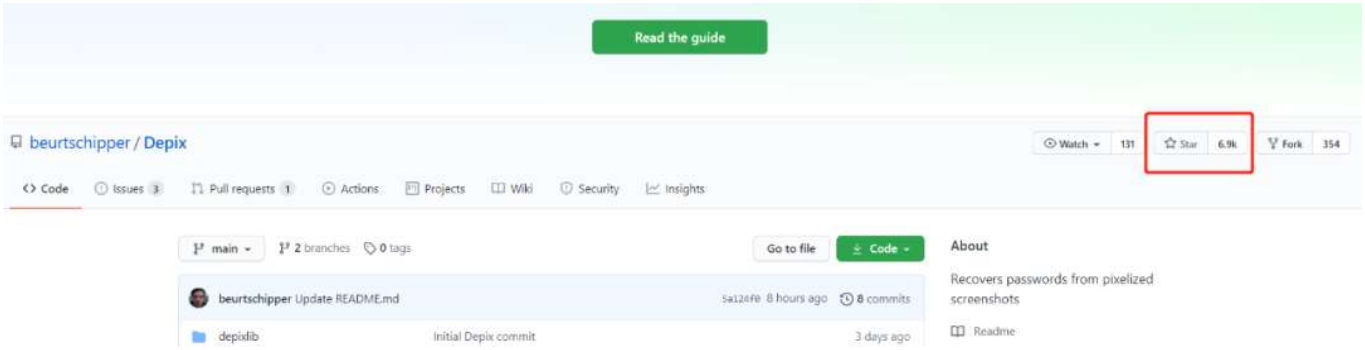
再比如：



看图找马赛克！（找不到请看右侧原图）

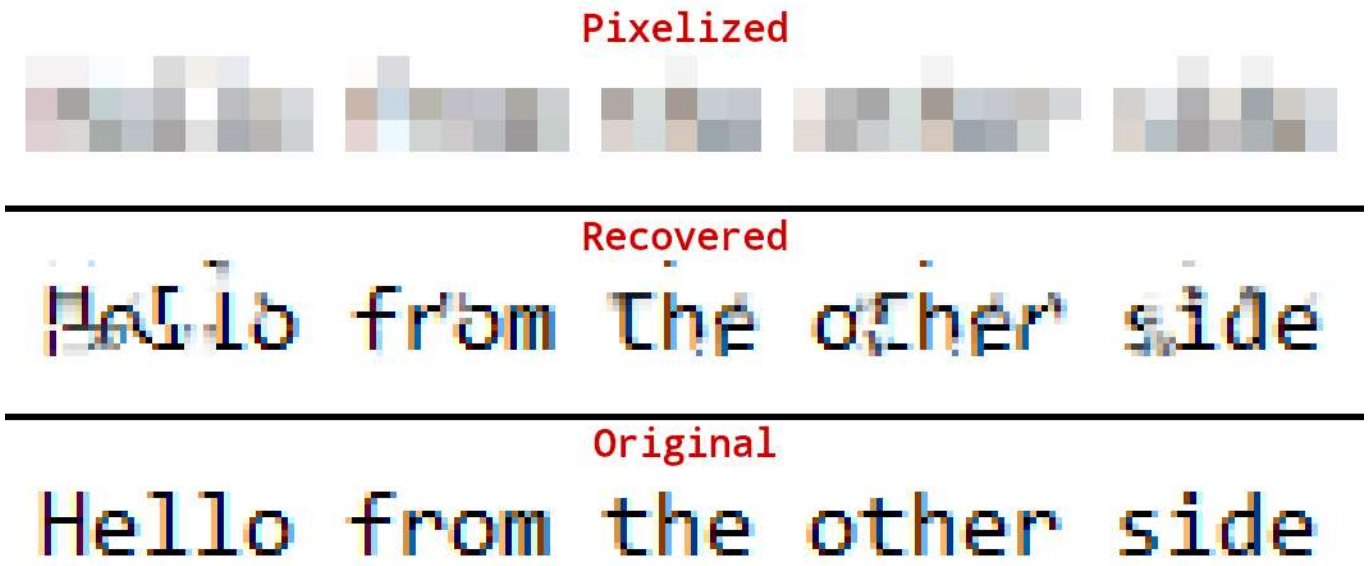
但是，在你想隐藏信息的同时，有一些技术却反其道而行之，试图将图片还原为原始状态。

最近，一个名为 Depix 的 GitHub 项目爆火，上线三天 star 量已经高达 6.9k。项目作者 Sipke Mellema 是一名信息安全顾问。



项目地址：https://github.com/beurtschipper/Depix

Depix 能够从像素化图像截图中恢复原图中包含的文字密码。该项目适用于使用线性方框滤波器（linear box filter）创建的像素化图像。如下图所示，项目作者给出了像素化图像、恢复之后的效果和原图的对比结果：



马赛克打得够严实了，不过 Depix 还是基本解读出了被隐藏的信息。

如何使用

使用 Depix 从像素化图像截图中恢复文字密码，操作也比较简单：

- 从截图中分割出矩形像素化 block；
- 在具有相同字体设置（包括文本大小、字体、颜色、hsl）的编辑器中，粘贴待处理字符的德布鲁因（De Bruijn sequence）。
- 给该序列截图，尽可能使用和像素化图像相同的截图工具。
- 执行命令：

```
python depix.py -p [pixelated rectangle image] -s [search sequence image] -o out
```

Depix 算法利用线性方框滤波器单独处理每一个 block 这一事实。它对搜索图像中的每一个 block 执行像素化以寻找直接匹配。

对于大部分像素化图像，Depix 尽量寻找单匹配结果，并假设这些匹配是正确的。至于周围多匹配 block 的结果被看作像素化图像中相同的几何距离，并认为这些匹配也是正确的。该过程重复多次。

在正确的 block 没有更多几何匹配后，Depix 将直接输出所有正确的 block。对于多匹配 block，Depix 将输出所有匹配的平均值。

Depix 背后的算法

像素化常使用线性方框滤波器实现。线性方框滤波器的实现很简单，速度很快，可以并行处理多个 block。

由于线性方框滤波器是一种确定性算法，对同样的值执行像素化通常会产生同样的像素化 block。使用同样位置的 block 对相同文本执行像素化，会得到同样的 block 值。我们可以尝试像素化文本来找出匹配的模式。幸运的是，这对于秘密值的一部分同样奏效。我们可以把每个 block 或 block 组合看作一个子问题。

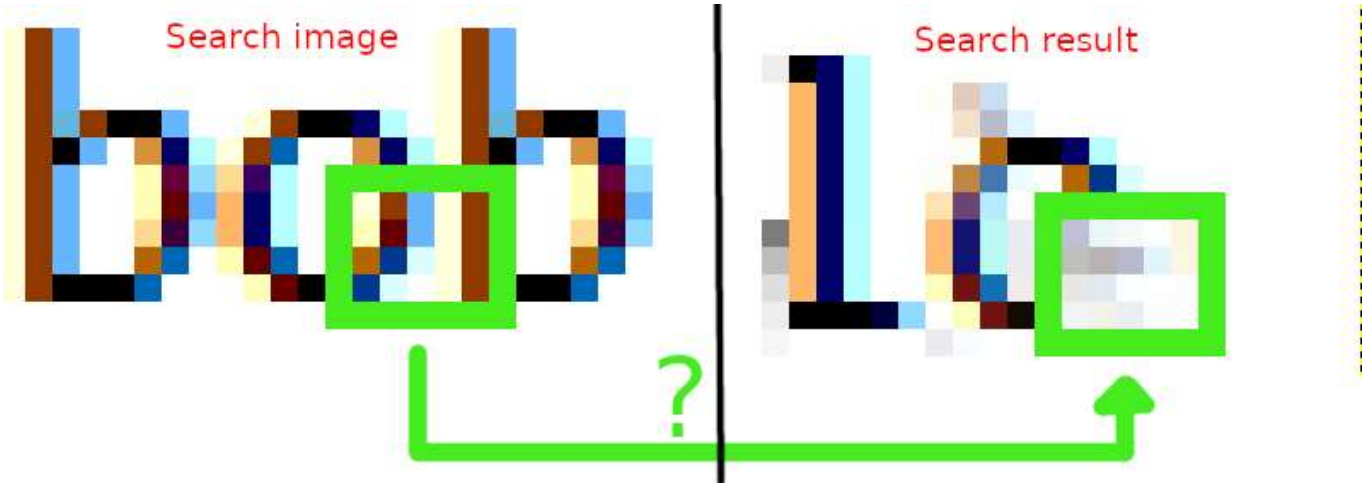
项目作者没有选择创建潜在字体的查找表。该算法要求在相同背景上具备相同的文本大小和颜色。现代文本编辑器还会添加色调、饱和度和亮度，也就是说存在海量潜在字体。

项目作者给出的解决方案也很简单：使用待处理字符的德布鲁因序列，将其粘贴到相同的编辑器中，然后截图。该截图可以用作相似 block 的查找图像，例如：

```
0010203040506070809 0a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0p0q0r0s0t0u0v0w0x0y0z0A0B0C0D0E0F0G0H0I0J0K0L0M0N0O0P0Q0R0S0T0U0V0W0X0Y0Z1112131415161718191a1b1c1
d1e1f1g1h1i1j1k1l1m1n1o1p1q1r1s1t1u1v1w1x1y1z1A1B1C1D1E1F1G1H1I1J1K1L1M1N1O1P1Q1R1S1T1U1V1W1X1Y1Z2232425262728292a2b2c2d2e2f2g2h2i2j2k2l2m2n2o2p2q2
r2s2t2u2v2w2x2y2z2A2B2C2D2E2F2G2H2I2J2K2L2M2N2O2P2Q2R2S2T2U2V2W2X2Y2Z3343536373839 3a3b3c3d3e3f3g3h3i3j3k313m3n3o3p3q3r3s3t3u3v3w3x3y3z3A3B3C3D3E3F3
G3H3I3J3K3L3M3N3O3P3Q3R3S3T3U3V3W3X3Y3Z4454647484 94a4b4c4d4e4f4g4h4i4j4k414m4n4o4p4q4r4s4t4u4v4w4x4y4z4A4B4C4D4E4F4G4H4I4J4K4L4M4N4O4P4Q4R4S4T4U4V4
W4X4Y4Z55665758595a5b5c5d5e5f5g5h5i5j5k515m5n5o5p5q5r5s5t5u5v5w5x5y5z5A5B5C5D5E5F5G5H5I5J5K5L5M5N5O5P5Q5R5S5T5U5V5W5X5Y5Z66768696a6b6c6d6e6f6g6h6i6j
6k616m6n6o6p6q6r6s6t6u6v6w6x6y6z6A6B6C6D6E6F6G6H6I6J6K6L6M6N6O6P6Q6R6S6T6U6V6W6X6Y6Z778797a7b7c7d7e7f7g7h7i7j7k717m7n7o7p7q7r7s7t7u7v7w7x7y7z7A7B7C
7D7E7F7G7H7I7J7K7L7M7N7O7P7Q7R7S7T7U7V7W7X7Y7Z8898a8b8c8d8e8f8g8h8i8j8k818m8n8o8p8q8r8s8t8u8v8w8x8y8z8A8B8C8D8E8F8G8H8I8J8K8L8M8N8O8P8Q8R8S8T8U8V8W
8X8Y8Z99a9b9c9d9e9f9g9h9i9j9k919m9n9o9p9q9r9s9t9u9v9w9x9y9z9A9B9C9D9E9F9G9H9I9J9K9L9M9N9O9P9Q9R9S9T9U9V9W9X9Y9Zaabacadaeafagahaiajakalamanaoapaqara
satauavawaxayazaAaBaCaDaEaFaGaHaIaJaKaLaMaNaOaPaQaRaSaTaUaVaWaXaYaZbbcbdbebfbgbhbibjbkblbmbnbobpbqbrbsbtbubvbwbxbybzbAb8bCbDbEbFbGbHbIbJbKbLbMbNbOb
PbQbRbSbTbUbVbWbXbYbZccdcecfcgchcicjckclcmcncocpcqcrcsctcucvcwcxcyczcAcBcCcDcEcFcGcHcIcJcKcLcMcNcOcPcQcRcScTcUcVcWcXcYcZddedfdgdhdidjdkdldmdndodpdq
drdsdtdudvdwdxdydzdAdBdCdDdEdFdGdHdIdJdKdLdMdNdOdPdQdRdSdTdUdVdWdXdYdZeefegeheiejekelemeneoepeqereseteuevewexeyezeAeBeCeDeEeFeGeHeIeJeKeLeMeNeOePeQ
eReSeTeUeVeWeXeYeZffgfhfifjfkflfmfnfofpfqfrfsftfufvfwfxfyfzfAfBfCfDfEfFfGfHfIfJfKfLfMfNfOfPfQfRfSfTfUfVfWfXfYfZgghgigjgkglgmgngogpgqgrgsgtgugvgwgxg
ygzgAgBgCgDgEgFgGgHgIgJgKgLgMgNgOgPgQgRgSgTgUgVgWgXgYgZhhihjhkhlhmhnhohphqhrhshthuhvhwhxhyhzhAhBhChDhEhFhGhHhIhJhKhLhMhNhOhPhQhRhShThUhVhWhXhYhZiij
ikiliminioipiqirisitiuiviwixiyiziAiBiCiDiEiFiGiHiIiJiKiLiMiNiOiPiQiRiSiTiUiViWiXiYiZjjkjljmjnjojpjqjrjsjtjujvjwjxjyjzjAjBjCjDjEjFjGjHjIjJjKjLjMjNjO
jPjQjRjSjTjUjVjWjXjYjZkklkmknkokpkqkrksktkukvkwkxkykzkAkBkCkDkEkFkGkHkIkJkKkLkMkNkOkPkQkRkSkTkUkVkWkXkYkZllmlnlolplqlrls1tlulvlwlxlylzlAlBlClDlElFl
GlHlIlJlKlLlMlNlOlPlQlRlSlTlUlVlWlXlYlZmmnmompmqmrmsmtmumvmwmxmymzmAmBmCmDmEmFmGmHmImJmKmLmMmNmOmPmQmRmSmTmUmVmWmXmYmZnnonpnqnrnsntnunvnwnxnynznAnB
nCnDnEnFnGnHnInJnKnLnMnNnOnPnQnRnSnTnUnVnWnXnYnZoopoqorosotouovowoxoyozoAoBoCoDoEoFoGoHoIoJoKoLoMoNoOoPoQoRoSoToUoVoWoXoYoZppqprpsptpupvpwpxpypzpAp
BpCpDpEpFpGpHpIpJpKpLpMpNpOpPpQpRpSpTpUpVpWpXpYpZqqrqsqtquqvqwqxqyqzqAqBqCqDqEqFqGqHqIqJqKqLqMqNqOqPqQqRqSqTqUqVqWqXqYqZrrsrtrurvrwrxryrzrArBrCrDrE
rFrGrHrIrJrKrLrMrNrOrPrQrRrSrTrUrVrWrXrYrZsstsusvswsxsyszsAsBsCsDsEsFsGsHsIsJsKsLsMsNsOsPsQsRsSsTsUsVsWsXsYsZttutvtwtxtytztAtBtCtDtEtFtGtHtItJtKtLt
MtNtOtPtQtRtStTtUtVtWtXtYtZuuvuwuxuyuzuAuBuCuDuEuFuGuHuIuJuKuLuMuNuOuPuQuRuSuTuUuVuWuXuYuZvvwvxvyvzvAvBvCvDvEvFvGvHvIvJvKvLvMvNvOvPvQvRvSvTvUvVvWvX
vYvZwwxwywzwAwBwCwDwEwFwGwHwIwJwKwLwMwNwOwPwQwRwSwTwUwVwWwXwYwZxxyxzxAxBxCxDxExFxGxHxIxJxKxLxMxNxOxPxQxRxSxTxUxVxWxXxYxZyyzyAyByCyDyEyFyGyHyIyJyKyL
yMyNyOyPyQyRySyTyUyVyWyXyYyZzzAzBzCzDzEzFzGzHzIzJzKzLzMzNzOzPzQzRzSzTzUzVzWzXzYzZAABACADAEAFAGAHAIAJAKALAMANAOAPAQARASATAUAVAWAXAYAZBBCBDBEBFBGBHBI
BJBKBLBMBNBOBPBQBRBSBTBUBVBWBXBYBZCCDCECFCGCHCICJCKCLCMCNCOCPCQCRCSCTCUCVCWCXCYCZDDEDFDGDHDIDJDKDLDMDNDODPDQDRDSDTDUDVDWDXDYDZEEFEGEHEIEJEKELEMENEO
EPEQERESETEUEVEWEXEYEZFFGFHFIFJFKFLFMFNFOFPFQFRFSFTFUFVFWFXFYFZGGHGIGJGKGLGMGNGOGPGQGRGSGTGUGVGWGXGYGZHHIHJHKHLHMHNHOHPHQHRHSHTHUHVHWHXHYHZIIJIKILI
MINIOIPIQIRISITIUIVIWIXIYIZJJKJLJMJNJOJPJQJRJSJTJUJVJWJXJYJZKKLKMKNKOKPKQKRKSKTKUKVKWKXKYKZLLMLNLOLPLQLRLSLTLULVLWLXLYLZMMNMOMPMQMRMSMTMUMVMWMXMYMZ
NNONPNQNRNSNTNUNVNWNXNYNZOOPOQORO SOTOUOVOWOXOYOZPPQPRPSPTPUPVPWPXPYPZQQRQSQTQUQVQWQXQYQZRRSRTRURVRWRXRYRZSSTSUSVSWSXSYSZTTUTVTWTXTYTZUUVUWUXUYUZVVW
VXVYVZWWXWYWZXXYXZYYZZ0
```


德布鲁因序列包括待处理字符的所有双字符组合。这很重要，因为一些 block 会重叠两个字符。找出恰当的匹配需要搜索图像中具备相同像素配置的 block。

在以下测试图像中，Depix 算法无法找到「o」的一部分。这是因为在搜索图像中，搜索 block 还包含下一个字母（「d」）的一部分，但在原始图像中这里有个空格。



创建字母的德布鲁因序列时加上空格显然会带来同样的问题：算法无法找到后续字母的恰当 block。有空格又有字母的图像需要更长的搜索时间，但结果也更好。

对于大部分像素化图像而言，Depix 似乎能够找到 block 的单匹配结果，并假设这是正确的。然后将其周围多匹配 block 的匹配结果看作在像素化图像中处于相同的几何距离，并假设这些匹配也是正确的。

在正确的 block 没有更多几何匹配后，Depix 直接输出所有正确的 block。对于多匹配 block，Depix 将输出所有匹配的平均值。虽然 Depix 的输出并不完美，但已经算不错了。