为什么重要：当前，使用手机和单反摄影已经成为人们生活中不可或缺的一部分，如此大的图片存储量，对于云端存储的空间挑战性极大。本研究提出一种全新的JPEG云端存储方案，为云端存储服务提供商节省存储空间。

相对于其他文件类型，例如视频、安装程序等，照片在用户之间的重复率极低。如此大的独立数据模块，对云存储公司的空间开销极大。

Compared with other file types, such as videos and installers, the repeatability between users is high. Cloud storage companies only need to keep one copy. But most of the photos are personal to individuals. Such a large independent data module will cause a huge space overhead for cloud storage companies.

Why it important(可修改): At present, the use of mobile phones and SLR photography has become an indispensable part of people's lives. Such a large amount of picture storage is extremely challenging for cloud storage space. This research proposes a new JPEG cloud storage solution to save storage space for cloud storage service providers.

为什么是最新技术：通常来说，现阶段的去重研究是针对于所有类型的文件，本研究基于JPEG文件的特殊性，提出先抓住重新构造Huffman表节省部分存储空间，再进行传统的去重工作的方案。针对于JPEG文件而言，现阶段的压缩方式都会损坏图片的部分细节，本研究达到了节省存储空间的同时，不对用户的JPEG文件构成破坏。

Why is state-of-the-art（可修改）: Generally, the current deduplication research is aimed at all types of files. Based on the particularity of JPEG files, this research proposes to first grasp the reconstruction of the Huffman table to save some storage space, and then perform traditional dedup work program. For JPEG files, the current compression method will damage some details of the picture. In this study, the storage space will be saved without damaging the user's JPEG file.

初步设计

阶段一(JPEG的解码和编码)：

1. 提取目录下每个JPEG文件的4张哈夫曼表。
2. 构成4张大的哈夫曼表(NewHuffman)
3. 将图片原有的压缩数据根据NewHuffman重新压缩存储为NewData

阶段二(去重阶段)：

1. 将NewData分为小的区块
2. 构建出目录下所有JPEG文件的区块的索引

3.对区块进行去重处理，并更新索引，最后服务器中只留下相同区块的一个副本

initial design

Phase one (decoding and encoding of JPEG):

1. Extract 4 Huffman tables for each JPEG file in the directory.

2. Make up 4 large Huffman tables(NewHuffman)

3. Recompress the original compressed data of the picture according to NewHuffman and store it as NewData

Phase two (deduplication phase):

1. Divide NewData into small blocks

2. Build an index of the blocks of all JPEG files in the directory

3. Deduplicate the block and update the index, and finally only one copy of the same block is left in the server

~~初步设计~~

~~阶段一(JPEG的解码和编码)：~~

1. ~~对JPEG文件进行解码，提取每个JPEG文件的4张哈夫曼表(DHT区段)，分别为(DC 0/1 AC 0/1)。将JPEG压缩图像数据(SOS区段)解码到上一层次，即经过量化处理后的数据(dataBFhuffman)。~~
2. ~~使用目录下的所有的JPEG文件的哈夫曼表重新构成4张新的的哈夫曼表(NewHuffman)~~
3. ~~将dataBFhuffman根据NewHuffman重新压缩，称为NewData并存储回原本JPEG文件中。~~
4. ~~而NewHuffman则存储在文件目录下~~

~~阶段二(去重阶段)：~~

1. ~~将NewData分为小的区块，此处可调节，初步定为16\*16像素的区块。~~
2. ~~构建出目录下所有JPEG文件的区块的索引~~
3. ~~计算出每个区块的哈希值，根据哈希值对区块进行去重处理，并更新索引，最后服务器中只留下相同区块的一个副本~~
4. ~~最后将索引写回到每个JPEG文件中，并将去重后的区块放置于文件目录下下。~~

~~initial design~~

~~Phase one (decoding and encoding of JPEG):~~

~~1. Decode JPEG files, extract 4 Huffman tables (DHT sections) for each JPEG file, respectively (DC 0/1 AC 0/1). Decode the JPEG compressed image data (SOS section) to the previous level, which is the data (dataBFhuffman) after quantization processing.~~

~~2. Use the Huffman tables of all JPEG files in the directory to reconstruct 4 new Huffman tables (NewHuffman)~~

~~3. Recompress dataBFhuffman according to NewHuffman, call it NewData and store it back to the original JPEG file.~~

~~4. NewHuffman will be stored in the file directory~~

~~Phase two (deduplication phase):~~

~~1. Divide NewData into small blocks, which can be adjusted here, and initially set to 16 \* 16 pixel blocks.~~

~~2. Build an index of the blocks of all JPEG files in the directory~~

~~3. Calculate the hash value of each block, deduplicate the block according to the hash value, and update the index. Finally, only one copy of the same block will be left in the server.~~

~~4. Finally write the index back into each JPEG file and place the deduplicated blocks under the file directory.~~