



交互式图像分割与抠像

章国锋

浙江大学CAD&CG国家重点实验室

图像分割与抠像

图像分割指的是将数字图像中感兴趣的区域分割出来。



抠像除了要得到前景轮廓外还要进一步估计半透明通道，以实现真实的合成。



课程内容

一. 常用模型

- 高斯混合模型;
- 马尔科夫随机场;

二. 交互式图像分割算法

- GrabCut;
- Lazy Snapping;

三. 抠像算法

- 贝叶斯抠像
- Closed Form Matting;

四. 基于深度学习的图像分割

五. 基于深度学习的抠像

一. 高斯混合模型

高斯模型

高斯模型就是用高斯概率密度函数（正态分布曲线）精确地量化事物，将一个事物分解为若干的基于高斯概率密度函数（正态分布曲线）形成的模型。

高斯模型有单高斯模型（**SGM**）和混合高斯模型（**GMM**）两种。

单高斯模型(SGM)

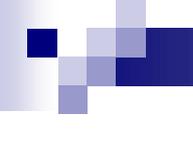
多维变量 x 服从高斯分布时，它的概率密度函数PDF为：

$$N(x;u,\Sigma)=\frac{1}{\sqrt{2\pi|\Sigma|}}\exp\left[-\frac{1}{2}(x-u)^T\Sigma^{-1}(x-u)\right] \quad \dots\dots(1)$$

x 是维度为 d 的列向量；

u 是模型期望；

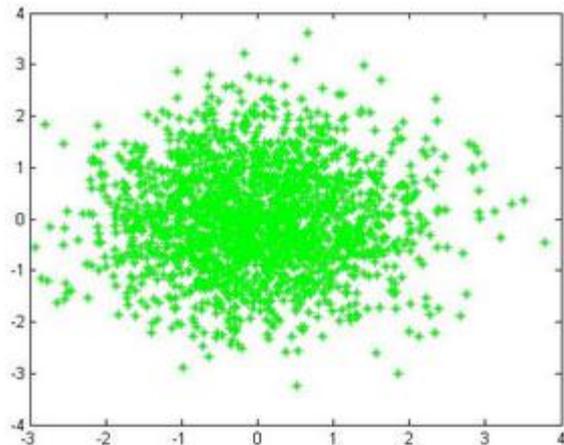
Σ 是模型方差。



在实际应用中， u 通常用样本均值来代替， Σ 通常用样本方差来代替。

很容易判断一个样本 x 是否属于类别C。因为每个类别都有自己的 u 和 Σ ，把 x 代入（1）式，当概率大于一定阈值时我们就认为 x 属于C类。

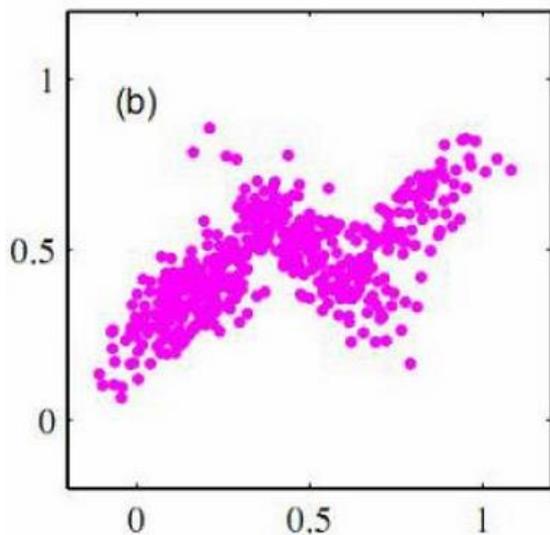
从几何上讲，单高斯分布模型在二维空间应该近似于椭圆，在三维空间上近似于椭球。



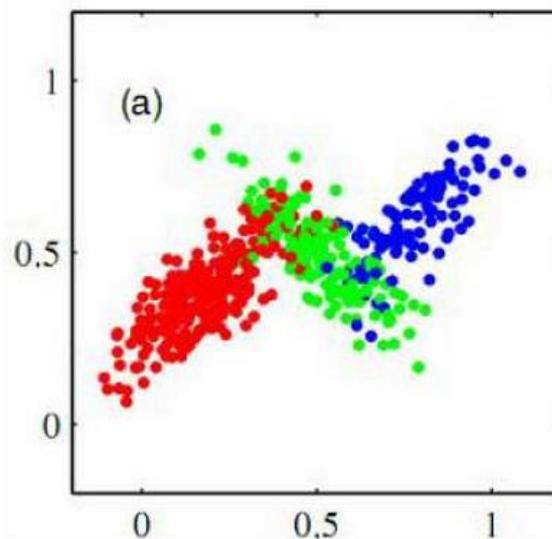
二维单高斯模型分布情况

高斯混合模型(GMM)

在很多分类问题中，属于同一类别的样本点并不满足“椭圆”分布的特性。这就引入了高斯混合模型。

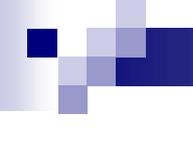


(a)



(b)

高斯混合模型图示，(a)表示所有样本数据；(b)表示已经明确了样本的分类



对于**(b)**图所示的情况，很明显，单高斯模型是无法解决的。为了解决这个问题，人们提出了高斯混合模型（**GMM**），顾名思义，就是数据可以看作是从数个高斯分布中生成出来的。

GMM（**Gaussian Mixture Model**），高斯混合模型（或者混合高斯模型），也可以简写为**MOG**（**Mixture of Gaussian**）。

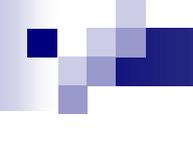
GMM认为数据是从几个GSM中生成出来的，即

$$Pr(x) = \sum_{k=1}^K \pi_k N(x; u_k, \Sigma_k) \quad \dots\dots(2)$$

其中的任意一个高斯分布 $N(x; u_k, \Sigma_k)$ 叫作这个模型的一个component;

K 需要事先确定好，代表component个数;

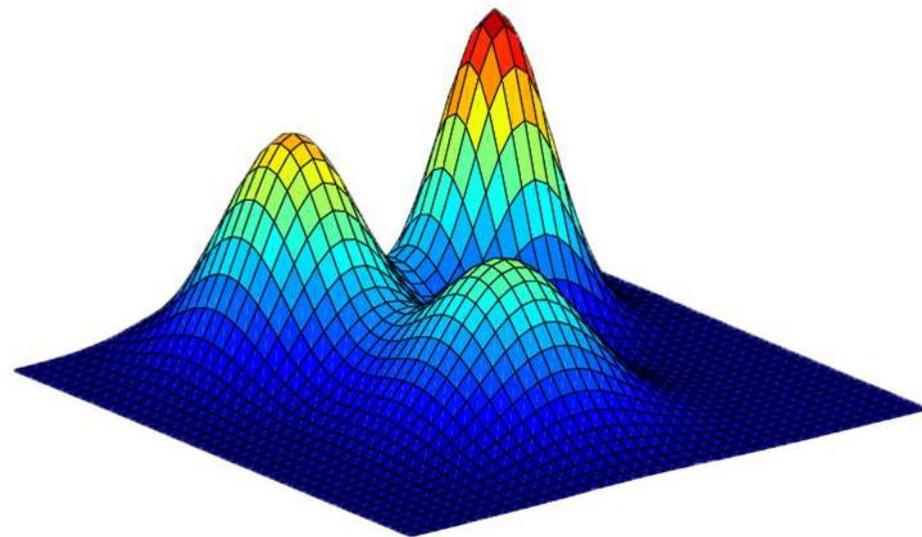
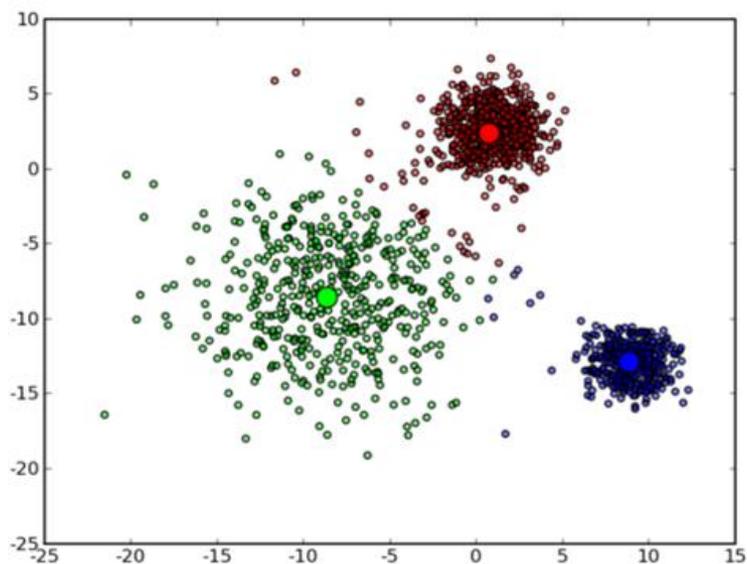
π_k 是权值系数.



每个 GMM 由 K 个 Gaussian 分布组成，每个 Gaussian 称为一个 “Component”，这些 Component 线性加成在一起就组成了 GMM 的概率密度函数。

当 K 足够大时，GMM 可以用来逼近任意连续的概率密度分布。

-将 K 个高斯模型混合在一起，每个点出现的概率是几个高斯混合的结果。



GMM是一种聚类算法，每个component就是一个聚类中心。即在只有样本点，不知道样本分类（含有隐含变量）的情况下，计算出模型参数（ π ， μ 和 Σ ）-----这显然可以用EM算法来求解。

再用训练好的模型去检查样本所属分类。
方法是：

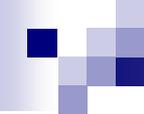
Step1:随机选择K个component中的一个（被选中的概率是 π_k ）；

Step2:把样本代入刚选好的component，判断是否属于这个类别，如果不属于则回到step1。

样本分类已知情况下的GMM

当每个样本所属分类已知时，GMM的参数非常好确定，直接利用Maximum Likelihood。设样本容量为N，属于K个分类的样本数量分别是 N_1, N_2, \dots, N_k ，属于第k个分类的样本集合是 $L(k)$ 。

$$\pi_k = \frac{N_k}{N} \quad \dots\dots(3)$$


$$u_k = \frac{1}{N_k} \sum_{x \in L(k)} x \quad \dots\dots(4)$$

$$\Sigma_k = \frac{1}{N_k} \sum_{x \in L(k)} (x - u_k) (x - u_k)^T \quad \dots\dots(5)$$

样本分类未知情况下的GMM

有N个数据点，服从某种分布 $\text{Pr}(x; \theta)$ ，我们想找到一组参数 θ ，使得生成这些数据点的概率最大，这个概率就是

$$\prod_{i=1}^N \text{Pr}(x_i; \theta)$$

称为似然函数（Likelihood Function）

通常单个点的概率很小，连乘之后数据会更小，容易造成浮点数下溢，所以一般取其对数，变成

$$\sum_{i=1}^N \log Pr(x_i; \theta)$$

称为log-likelihood function(对数似然函数)

GMM的log-likelihood function就是：

$$\sum_{i=1}^N \log \left\{ \sum_{k=1}^K \pi_k N(x_i; \mu_k, \Sigma_k) \right\} \quad \dots\dots(6)$$

这里每个样本 x_i 所属的类别 z_k 是不知道的。 Z 是隐含变量；

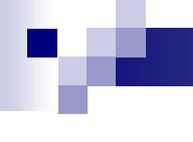
我们就是要找到最佳的模型参数，使得(6)式所示的期望最大，“期望最大化算法”名字由此而来。

EM (Expectation Maximum) 法求解

EM要求解的问题一般形式是：

$$\theta^* = \arg \max_{\theta} \prod_{j=1}^{|\mathbf{x}|} \sum_{y \in \mathcal{Y}} \Pr(X = x_j, Y = y; \theta)$$

Y是隐含变量。



EM算法的基本思路是：随机初始化一组参数 $\theta^{(0)}$ ，根据后验概率 $\Pr(Y|X;\theta)$ 来更新 Y 的期望 $E(Y)$ ，然后用 $E(Y)$ 代替 Y 求出新的模型参数 $\theta^{(1)}$ 。如此迭代直到 θ 趋于稳定。

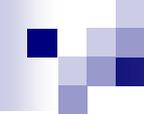
如果数据点的分类标签 Y 是已知的，那么求解模型参数直接利用Maximum Likelihood就可以了。

E-Step

E是Expectation的意思，就是假设模型参数已知的前提下求隐含变量Z分别取 z_1, z_2, \dots 的期望，亦即Z分别取 z_1, z_2, \dots 的概率。在GMM中就是求数据点由各个 component 生成的概率。

$$\gamma(i, k) = \alpha_k \cdot Pr(z_k | x_i; \pi, \mu, \Sigma) \quad \dots\dots(7)$$

注意到我们在Z的后验概率前面乘以了一个权值因子 α_k ，它表示在训练集中数据点属于类别 z_k 的频率，在GMM中它就是 π_k 。


$$\gamma(i,k) = \frac{\pi_k N(x_i | u_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(x_i | u_j, \Sigma_j)} \dots\dots(8)$$

M-Step

M就是Maximization的意思，就是用最大似然的方法求出模型参数。

现在我们认为上一步求出的 $\gamma(i,k)$ 就是“数据点 x_i 由 component k 生成的概率”。根据公式(3),(4),(5)可以推出：

$$N_k = \sum_{i=1}^N \gamma(i,k) \quad \dots\dots(9)$$

$$u_k = \frac{1}{N_k} \sum_{i=1}^N \gamma(i,k) x_i \quad \dots\dots(10)$$

$$\Sigma_k = \frac{1}{N_k} \sum_{i=1}^N \gamma(i,k) (x_i - u_k)(x_i - u_k)^T \quad \dots\dots(11)$$

$$\pi_k = \frac{N_k}{N} \quad \dots\dots(12)$$

二.马尔科夫随机场与图像处理

马尔科夫随机过程

通俗的讲，马尔科夫随机过程就是，下一个时间点的状态只与当前的状态有关系，而与以前的状态没有关系，即未来的状态决定于现在而不决定于过去。

一维马尔科夫过程

设有随机过程 $\{X_n, n \in T\}$, 若对于任意正整数 $n \in T$ 和任意的 $i_0, i_1, \dots, i_{n+1} \in I$, 条件概率满足

$$P\{X_{n+1} = i_{n+1} \mid X_0 = i_0, \dots, X_n = i_n\} = P\{X_{n+1} = i_{n+1} \mid X_n = i_n\}$$

就称 $\{X_n, n \in T\}$ 为马尔科夫过程, 该随机过程的统计特性完全由条件概率所决定

马尔科夫随机过程的分类

按照参数集和状态空间分成四类：

- 时间和状态都是离散的马尔科夫过程。也称为马尔科夫链；
- 时间连续、状态离散的马尔科夫过程。通常称为纯不连续马尔科夫过程；
- 时间和状态都是连续的马尔科夫过程；
- 时间离散、状态连续的马尔科夫过程。

马尔科夫随机场

马尔科夫随机场包含两层意思：

- 马尔科夫性质
- 随机场

马尔科夫性质

马尔科夫性质指的是一个随机变量序列按时间先后关系依次排开的时候，第 $N+1$ 时刻的分布特性，与 N 时刻以前的随机变量的取值无关。

拿天气来打个比方：如果我们假定天气是马尔可夫的，其意思就是我们假设今天的天气仅仅与昨天的天气存在概率上的关联，而与前天及前天以前的天气没有关系。其它如传染病和谣言的传播规律，就是马尔科夫的。

随机场

当给每一个位置中按照某种分布随机赋予相空间的一个值之后，其全体就叫做随机场。其中有两个概念：位置（**site**），相空间（**phase space**）。我们不妨拿种地来打个比方。“位置”好比是一亩亩农田；“相空间”好比是要种的各种庄稼。我们可以给不同的地种上不同的庄稼，这就好比给随机场的每个“位置”，赋予相空间里不同的值。所以，通俗点讲，随机场就是在哪块地里种什么庄稼的事情。

。

马尔科夫随机场

拿种地打比方，如果任何一块地里种的庄稼的种类仅仅与它邻近的地里种的庄稼的种类有关，与其它地方的庄稼的种类无关，那么这些地里种的庄稼的集合，就是一个马尔科夫随机场。

马尔科夫随机场与图像的关系

一维马尔科夫随机过程很好的描述了随机过程中某点的状态只与该点之前的一个点的状态有关系。对于定义在二维空间上的图像，也可以将它看为一个二维随机场。自然也存在二维马尔科夫随机场,此时必须考虑空间的关系,二维MRF的平面网格结构同样可以较好的表现图像中像素之间的空间相关性。

基本定义

设 $S = \{(i, j) | 1 \leq i \leq M, 1 \leq j \leq N\}$ 表示MN位置的有限格点集
即随机场中的位置

$\Lambda = \{1, 2, \dots, L\}$ 表示状态空间，即随机场中的相空间

$X = \{x_s | s \in S\}$ 表示定义在 $\forall s \in S$ 处的随机场

x_s 表示在随机场 X 上，状态空间为 Λ 的隐状态随机变量，
即 $x_s \in \Lambda$

在图像中：

- 格点集 \mathbf{S} 表示像素的位置；
- X 称为标号场，也可以表示像素值的集合或图像经小波变换后的小波系数集合；
- Λ 为标号随机变量 x_s 的集合；
- L 表示将图像分割为不同区域的数目。

邻域系统

设 $\delta = \{ \delta(s) \mid s \in S \}$ 是定义在 S 上的通用邻域系统的集合，其满足如下特性：

$$(1) \delta(s) \subset S$$

$$(2) s \notin \delta(s)$$

$$(3) \forall s, r \in S, s \in \delta(r) \Leftrightarrow r \in \delta(s)$$

则位置 $r \in \delta(s)$ 称作 s 的邻点， $\delta(s)$ 称作 s 的邻点集

分阶邻域系统与子团

在图像模型中，可以根据对象元的距离建立一种分阶邻域系统，定义如下：

$\delta^{(n)}(s) = \{r \mid d(s, r) \leq n, r \neq s\}$, 式中 n 为邻域系统的阶次， $d(\bullet)$ 表示距离函数，经常使用欧氏距离，市区距离，棋盘距离等函数。

对 $\forall n \geq 0$, 满足特性 $\delta^{(n)}(s) \subset \delta^{(n+1)}(s)$

5	4	3	4	5
4	2	r	2	4
3	1	(i, j)	1	3
4	2	1	2	4
5	4	3	4	5

分阶邻域系统

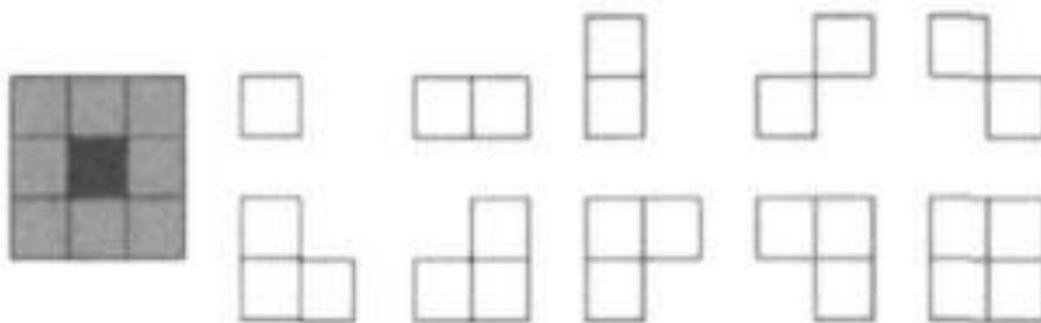
子团

S 中有不同的邻域结构，在 S 上由单个像元或由象元与其邻点组成的子集 $c \subset S$ 称为一个子团。子团 c 的集合用 C 来表示。

分阶邻域系统与子团示例



(a) 一阶邻域系统和原子团



(b) 二阶邻域系统和原子团

马尔科夫随机场

设 δ 为 S 上的邻域系统，若随机场 $X=\{x_s, s \in S\}$ 满足如下条件：

$$(1) P\{X=x\} > 0, \forall x \subset \Lambda$$

$$(2) P\{X_s = x_s \mid X_r = x_r, r \neq s, \forall r \in \delta(s)\}$$

$$= P\{X_s = x_s \mid X_r = x_r, \forall r \in \delta(s)\}$$

则称 X 为以 δ 为邻域系统的马尔科夫随机场，

上式称为马尔科夫随机场的局部特性。

邻域系统 δ 的MRF的含义：在任意格点 s 的其余格点位置上随机变量 x_s 取值已知的条件下，随机场在格点 s 处的取值概率只与格点 s 的 δ 相邻点有关。

在图像中， $P(\bullet)$ 表示标号场的先验概率， $P(\bullet|\bullet)$ 表示邻域系统标号的局部作用关系

在数字图像中，一个象元的灰度值仅与其邻域系统内各象元的灰度值有关，因而可以利用马尔科夫随机场来模拟数字图像。当邻域系统 δ 足够大时，任何定义在 \mathbf{S} 上的图像数据均可看成马尔科夫随机场的一个实现。

MRF与Gibbs分布的等价关系

由于标号场先验概率和标号场的邻域局部关系在实际应用中很难确定，20世纪80年代Hammersley-Clifford给出了Gibbs分布与MRF的关系，从而用Gibbs分布求解MRF中的概率分布。

δ 是定义在 S 上的邻域系统，当且仅当随机场 $X=\{x_s, s \in S\}$ 的联合概率分布具有如下形式：

$$P(X = x) = (1/Z)\exp\{-U(x)\}$$

则称 X 为吉布斯随机场。

式中 x 是随机场 X 的一“实现”，即 X 在格点集 S 上的一组态。

$$U(x) = - \sum_{c \in C} V_c(x) \text{ 称为能量函数,}$$

$V_c(x)$ 是仅与子团 c 内各象元值有关的子团势函数。

$$Z = \sum_x e^{-U(x)} \text{ 称为配分函数, 是一个归一化常数。}$$

Gibbs分布与MRF的等价条件：一个随机场是关于邻域系统的**MRF**，当且仅当这个随机场是关于邻域系统的**Gibbs分布**，表示为：

$$P(x_s | x_r, r \in \delta(s)) = \frac{\exp \left[- \left(\sum_{c \in C} V_c(x_s | x_r) \right) \right]}{\sum_{x_s=1}^L \exp \left[- \left(\sum_{c \in C} V_c(x_s | x_r) \right) \right]}$$

上式解决了求MRF中概率分布的难题,使对MRF的研究转化为对势函数 $V_c(\mathbf{x})$ 的研究,使Gibbs分布与能量函数建立了等价关系,是研究邻域系统 $\delta(\mathbf{s})$ MRF的一个重要里程碑。

基于MRF的图像分割模型

基于马尔科夫随机场模型的图像分割算法假设待分割图像的像素只与其邻域内的像素相关，与邻域外的像素无关；基于该假设我们能定量计算图像局部的先验结构信息，并根据最大后验概率准则（MAP），有效的利用像素间结构信息分割图像。

对于一幅给定的 $M*N$ 的图像 Y ，
其中任意一个像素 y_i ，分割后对应的标记为 x_i ，
定义两个随机场：

$X=\{x_i, i \in S\}$ 是图像分割后的类别标号场，
 $x_i = 1, 2, \dots, L$ 表示分割成 L 个区域，
但其类别状态不能直接观察到。

$Y=\{y_i, i \in S\}$ 是可观测的随机场，
即图像的观测灰度场，
那么分割问题可以描述为：

$$y_i \in Y \rightarrow x_i = \begin{cases} 1 & i = t \arg et1 \\ 2 & i = t \arg et2 \\ \vdots & \vdots \\ L & i = t \arg etL \end{cases} \quad x_i \in X, i \in \{1, 2, \dots, M * N\}$$

根据贝叶斯准则，最优分割准则为：

$$\hat{X} = \arg \max_X p(X | Y) = \arg \max_X \frac{p(Y | X)p(X)}{p(Y)}$$

对于一幅给定的图像，Y已知，所以 $p(Y)$ 为常数，

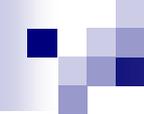
故上式等价于： $\arg \max_X p(Y | X) p(X)$

由于随机场X是MRF，具有正概率性和Markov性。

由MRF与Gibbs分布的等价性可知

$$p(X) = \prod_{i=1}^{M*N} p(x_i) = \frac{\prod_{i=1}^{M*N} \exp\left(-\sum_{c \in C} V_c(x_i)\right)}{\sum_{x_i=1}^L \exp\left(-\sum_{c \in C} V_c(x_i)\right)}$$

式中 $V_c(x_i)$ 是包含 x_i 的基团c的势函数，C是所有基团的集合

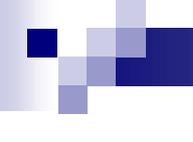

$$V_c(x_i) = \begin{cases} -\beta & \text{if } x_i = x_j \\ +\beta & \text{if } x_i \neq x_j \end{cases}$$

其中 β 为耦合系数

对于 $p(Y | X)$ 是条件概率函数，也称为似然函数，其概率分布要根据图像服从的分布而选择。

在这里我们假设图像的各个位置的像素是独立同分布的， $p(Y | X)$ 服从高斯分布，即：

$$p(Y=y | X=x) = \prod_s p(Y = y_s | X = x_s) = \prod_s \frac{1}{\sqrt{2\pi\sigma_m}} \exp\left[-\frac{(y_s - u_m)^2}{2\sigma_m^2}\right]$$



其中参数 u_m 和 σ_m 分别是第 m 区域的均值和标准差，可以根据其在概率论中的公式计算。

根据上面得到的 $p(x)$ 和 $p(Y|X)$ 的计算公式就可求得最优分割结果 \hat{X} 。

MRF图像分割的关键点

- 基团的选取
- 势函数 $V_c(x)$ 的定义
- 先验概率 $p(X)$ 的确定（图像预分割）
- 选取合适的条件分布
- 条件分布中参数的估计
- 分割算法的选取

图像分割的定义

对图像进行研究和应用时，人们往往对图像中某些部分感兴趣，这些部分通常被称为目标或对象（object）。图像分割是根据所要研究的对象，把图像分成互不重叠的区域，并提取出感兴趣的目标。

针对单色图像的分割。给定一个初始化的 Trimap T 。图像的索引大小为 n ，灰度阵列 $z = (z_1, \dots, z_n, \dots, z_N)$ 。图像在每个像素分割并被表示为一个“不透明度”值的阵列。通常 $0 \leq a_n \leq 1$ ，但对于硬分割， $a_n = \{0, 1\}$ ，0代表背景而1代表前景。系数 $\underline{\theta}$ 描述图像的前景和背景灰度水平分布组成的灰度直方图 (histograms of grey values):

$$\underline{\theta} = \{h(z; \alpha), \alpha = 0, 1\}$$

一个对应背景,一个对应前景。灰度直方图可直接由相应的Trimap区域 T_B, T_F 上的标记元素获得。

(灰度直方图在灰度区域被正态化为总和为1: $\int_z h(z; \alpha) = 1$)。

分割任务是根据给定的图像数据 z 和模型 θ 推测未知透明度 α 。

依靠能量最小化的分割

定义一个能量函数 E ，其最小值对应理想的分割。由于前景和背景灰度水平直方图和不透明度是连贯的，并能反映物体实体化的倾向。这样获得一个“吉布斯 (Gibbs)”能量的形式：

$$E(\underline{\alpha}, \underline{\theta}, z) = U(\underline{\alpha}, \underline{\theta}, z) + V(\underline{\alpha}, z)$$

数据项 U 估计了不透明度分布 $\underline{\alpha}$ 对图像数据 z 的符合程度。给定直方图模型 $\underline{\theta}$ ，则 U 可定义为：

$$U(\underline{\alpha}, \underline{\theta}, z) = \sum_n -\log h(z_n; a_n)$$

平滑项 (smoothness term) 能够被写成：

$$V(\underline{\alpha}, z) = \gamma \sum_{(m,n) \in C} dis(m,n)^{-1} [a_n \neq a_m] \exp -\beta (z_m - z_n)^2$$

此处 $[\phi]$ 对于可预测 ϕ 是一个取值为0,1的指示器函数， C 是一系列邻居像素对， $dis(\cdot)$ 为邻居像素的欧几里得距离。这个能量函数能够促使相似灰度水平区域的连贯。在实际应用中，定义水平、垂直、斜角相邻的像素（8连通方式）能够获得好的结果。

八邻域

取像素 p 四周8个点作为相邻接的邻域点，称为像素 p 的八邻域。在下图中，除掉 p 本身 (x,y) 外，剩下的8个点就是 p 的八邻域。互为八邻域的像素又被称为八连通的。四邻域与此类似。

$(x - 1, y - 1)$	$(x, y - 1)$	$(x + 1, y - 1)$
$(x - 1, y)$	(x, y)	$(x + 1, y)$
$(x - 1, y + 1)$	$(x, y + 1)$	$(x + 1, y + 1)$

若常量 $\beta=0$ 平滑项就是著名的伊辛优先 (Ising prior) 模型, 在一定程度上其由常数 γ 决定, 能够在各处趋向于平滑。

常数 β 在图像分割中被设置为:

$$\beta = \left(\left\langle (z_m - z_n)^2 \right\rangle \right)^{-1}$$

$\langle \cdot \rangle$ 表示一个图像采样的期望

现在就定义了整个能量模型，分割能够被看做一个全局最小化问题：

$$\hat{\underline{\alpha}} = \arg \min_{\underline{\alpha}} E(\underline{\alpha}, \underline{\theta})$$

运用一个标准的最小化cut算法获得最小值，这个算法形成了硬分割的基础。

三. GrabCut

GrabCut 课题背景

GrabCut是微软研究院的一个课题，主要功能是图像分割。

该算法利用了图像中的纹理（颜色）信息和边界（反差）信息，只要少量的用户交互操作即可得到比较好的分割结果。

GrabCut 与其他方法相比

Magic Wand

Intelligent Scissors

GrabCut

用户
输入



分割
结果



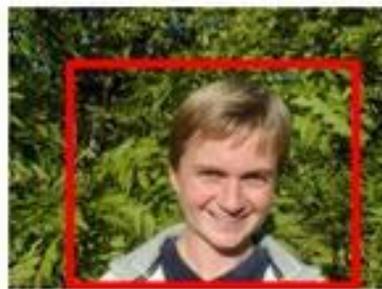
Regions

Boundary

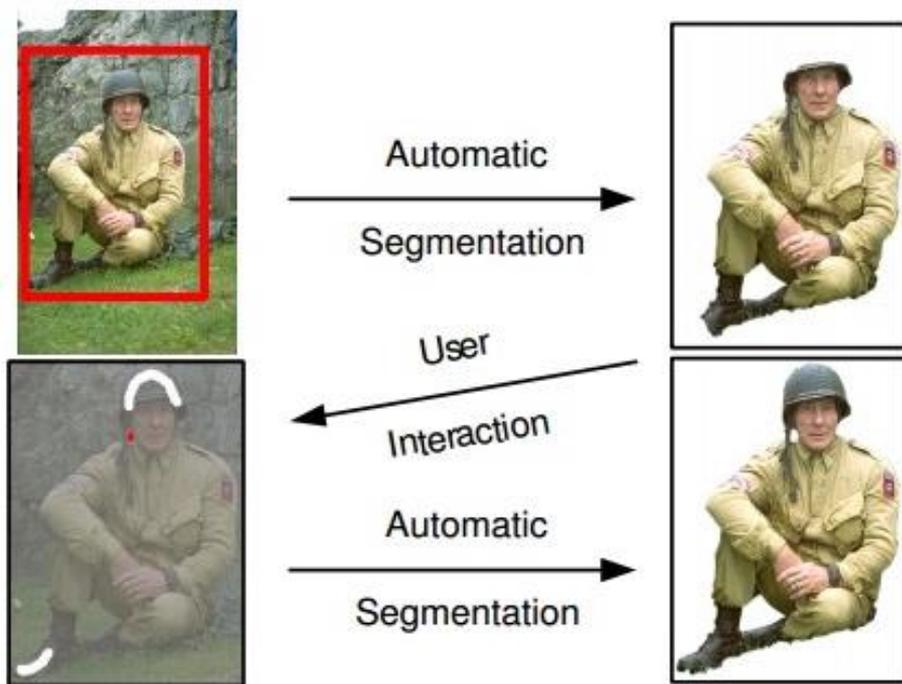
Regions & Boundary

简单的效果展示

(1) 用户只需要在目标外面画一个框，把目标框住，GrabCut就可以完成良好的分割：



(2) 如果增加额外的用户交互（由用户指定一些像素属于背景或前景），那么效果就会更好：

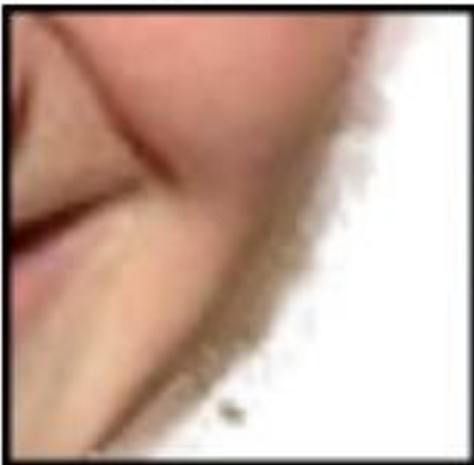


(3) 其Border Matting技术会使目标分割边界更加自然和平滑:

Knockout 2



Bayes Matte



GrabCut



GrabCut基于graph cuts实现

- 1.用高斯混合模型GMM(Gaussian Mixture Model)代替灰度直方图，颜色模型取代单色图像；
- 2.在graph cuts中只进行一次的最小化分割估计被一个在估计和参数学习中可选的、可靠迭代的过程代替；
- 对于Trimap，用户只需提供不完全标记。

算法原理

图像现在被看做是在**RGB**颜色空间内的组成像素 z_n 。由于构建足够的颜色空间直方图是不可行的，因此采用高斯混合模型**GMMs**。每个**GMM**是 K 个组件（**component**）的满协方差（**full-covariance**）高斯混合，通常 $K=5$ 。

为较好地处理**GMM**,在优化框架中引进一个额外的向量 $k = \{k_1, \dots, k_n, \dots, k_N\}$, 其中 $k_n \in \{1, \dots, K\}$ 。对应每个像素, 指定一个特殊的**GMM**组件, 这个**GMM**组件来源于背景或者前景模型, 分别对应着 $\alpha_n = 0$ 或者 $\alpha_n = 1$ 。

现在，对于分割的吉布斯能量公式可表示为：

$$E(\underline{\alpha}, k, \underline{\theta}, z) = U(\underline{\alpha}, k, \underline{\theta}, z) + V(\underline{\alpha}, z)$$

其也依赖于GMM组件中的变量 k ，考虑颜色GMM模型，数据项 U 可表示为：

$$U(\underline{\alpha}, k, \underline{\theta}, z) = \sum_n D(\alpha_n, k_n, \underline{\theta}, z_n)$$

此处

$$D(\alpha_n, k_n, \underline{\theta}, z_n) = -\log p(z_n | \alpha_n, k_n, \underline{\theta}) - \log \pi(\alpha_n, k_n)$$

$p(\cdot)$ 是一个高斯概率分布；

$\pi(\cdot)$ 是混合权重系数（等于常数）；

因此有：

$$D(\alpha_n, k_n, \underline{\theta}, z_n) = -\log \pi(\alpha_n, k_n) + \frac{1}{2} \log \det \Sigma(\alpha_n, k_n) \\ + \frac{1}{2} [z_n - \mu(\alpha_n, k_n)]^T \Sigma(\alpha_n, k_n)^{-1} [z_n - \mu(\alpha_n, k_n)]$$

模型的参数可表示为:

$$\underline{\theta} = \{ \pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k), \alpha = 0, 1, k = 1 \dots K \}$$

即, $2K$ 个高斯混合模型组件的权重 π , 均值 μ 和协方差 Σ 对应着背景和前景分布。

平滑项具体为:

$$V(\underline{\alpha}, z) = \gamma \sum_{(m,n) \in C} [\alpha_n \neq \alpha_m] \exp - \beta \| z_m - z_n \|^2$$

GrabCut进行迭代式的图像分割

1.初始化

- 用户通过只提供 T_B 初始化Trimap T 。前景设置为 $T_F = \text{非零值}$ ；设置 $T_U = \overline{T_B}$ ，完成背景设置。
- 对于 $n \in T_B$ 初始化 $\alpha_n = 0$ ，对于 $n \in T_U$ 初始化 $\alpha_n = 1$ 。
- 背景和前景高斯混合模型则由 $\alpha_n = 0$ 和 $\alpha_n = 1$ 对应初始化。

2. 迭代最小化

(1) 为像素指定高斯混合模型GMM组件:对于每个在 T_U 中的 n :

$$k_n := \arg \min_{k_n} D_n(\alpha_n, k_n, \theta, z_n)$$

(2) 从数据 z 中获得高斯混合模型GMM参数:

$$\underline{\theta} := \arg \min_{\underline{\theta}} U(\underline{\alpha}, k, \underline{\theta}, z)$$

(3) 估计分割：用最小cut求解：

$$\min_{\{\alpha_n : n \in T_U\}} \min_k E(\underline{\alpha}, k, \underline{\theta}, z)$$

(4) 重复步骤 (1)，直到收敛；

(5) 应用边缘修复。

3. 用户编辑

- 编辑：设定一些像素 $\alpha_n = 0$ （背景画刷）或者 $\alpha_0 = 1$ （前景画刷），相应的更新Trimap T 。执行上面的第（3）步一次。
- 修复操作：选择化执行整个迭代最小化算法。

不同方法结果比较



**Magic Wand
(198?)**

**Intelligent Scissors
Mortensen and
Barrett (1995)**

**Graph Cuts
Boykov and
Jolly (2001)**

**LazySnapping
Li et al. (2004)**

**GrabCut
Rother et al.
(2004)**

四. Lazy Snapping

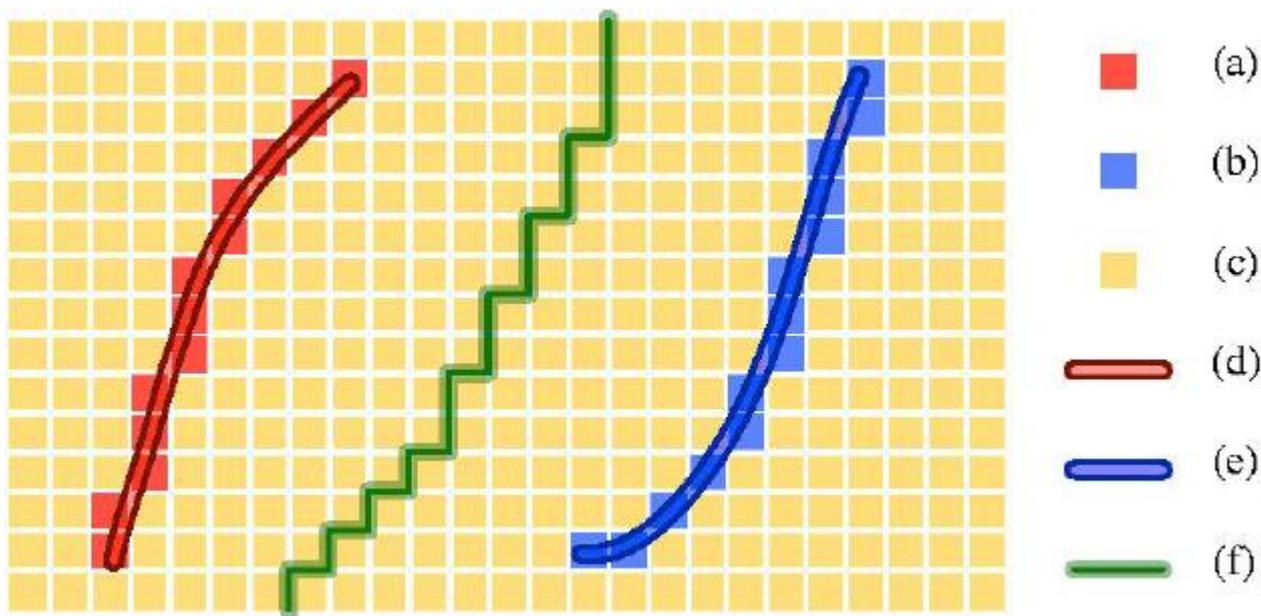
方法概览

目标: 从背景中分割出前景

- 第一步: 前景/背景选择
 - 只需少量的用户交互
 - 在segment level上的Mincut
- 第二步: 边缘修复
 - 在pixel level上的 Mincut

物体（前景）标记

- 快速分割前景和背景
- 在不同区域标记种子点
- 在pixel level上用graph cut algorithm 求解



graph cut 图像分割

能量最小化

$$E(X) = \sum_{i \in \mathcal{V}} E_1(x_i) + \lambda \sum_{(i,j) \in \mathcal{E}} E_2(x_i, x_j)$$

- 最小化吉布斯能量方程
 - $E_1(x_i)$: (区域项) Likelihood Energy
 - $E_2(x_i, x_j)$: (惩罚项) Prior Energy
- x_i : 节点*i*的label
 - {0: 背景, 1: 前景}

Likelihood Energy

$$E_1(x_i = 1) = 0 \quad E_1(x_i = 0) = \infty \quad \forall i \in \mathcal{F}$$

$$E_1(x_i = 1) = \infty \quad E_1(x_i = 0) = 0 \quad \forall i \in \mathcal{B}$$

$$E_1(x_i = 1) = \frac{d_i^{\mathcal{F}}}{d_i^{\mathcal{F}} + d_i^{\mathcal{B}}} \quad E_1(x_i = 0) = \frac{d_i^{\mathcal{B}}}{d_i^{\mathcal{F}} + d_i^{\mathcal{B}}} \quad \forall i \in \mathcal{U}$$

- 保证种子点的选取正确
- 对label不确定的pixel用colour similarity确定其energy.
 - $d_i^{\mathcal{F}}$: 距前景颜色的距离
 - $d_i^{\mathcal{B}}$: 距背景颜色的距离

Prior Energy

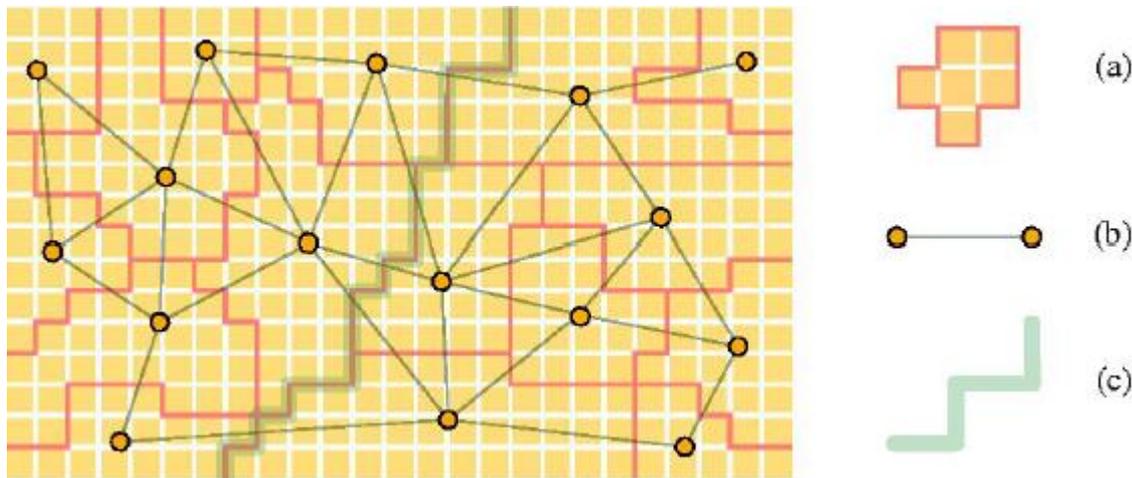
$$E_2(x_i, x_j) = |x_i - x_j| \cdot g(C_{ij})$$

$$C_{ij} = \|C(i) - C(j)\|^2$$

$$g(\epsilon) = \frac{1}{\epsilon + 1}$$

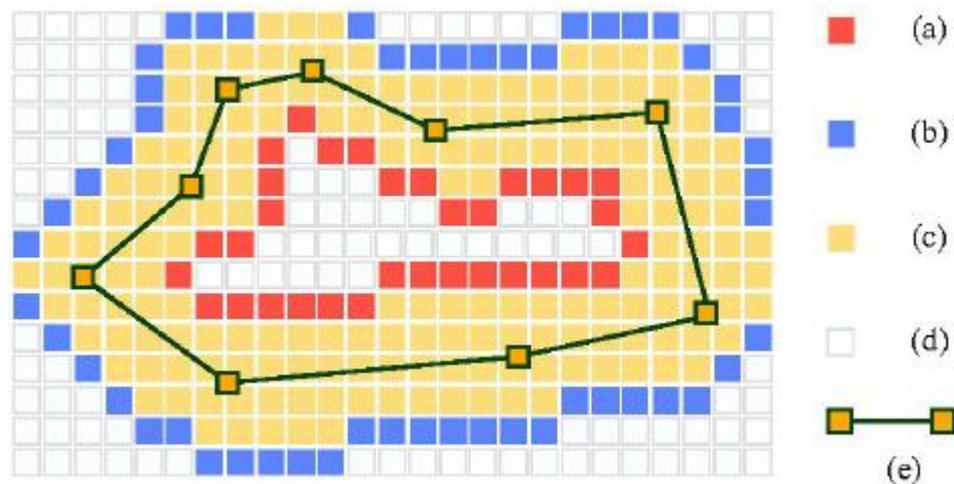
- 边界惩罚项
- 如果相邻pixel有相似的colour值，则值较大
- 只有横跨分割边界时，取非零值

分水岭算法 (*Watershed Algorithm*)



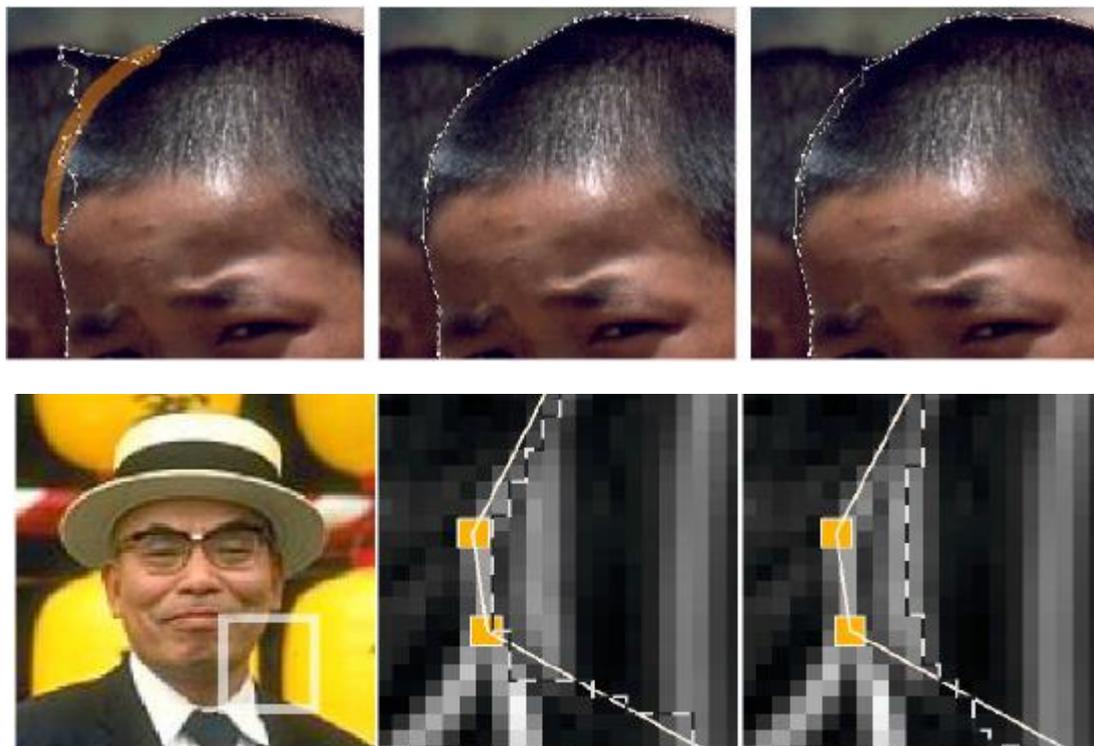
- 用分水岭算法对图像进行预分割
- graph cut 在segment level而非pixel level上执行
- 如果segment中的一个pixel相连接则连接两个segments

边缘修复 (*Boundary Editing*)



$$E_2(x_i, x_j) = |x_i - x_j| \cdot g \left((1 - \beta) \cdot C_{ij} + \beta \cdot \eta \cdot g(D_{ij}^2) \right)$$

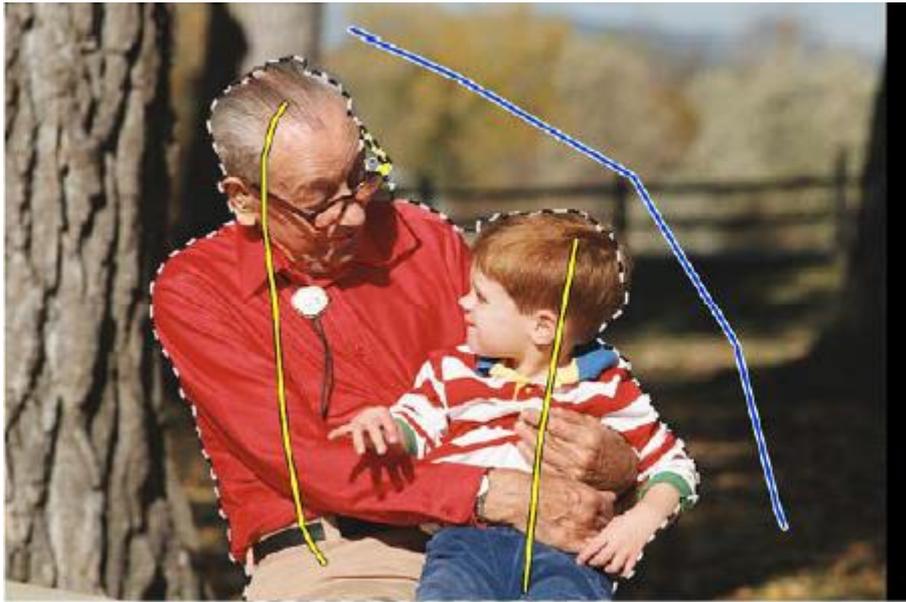
- Likelihood energy 和第一步分割相同
- 用多边形位置作为软约束增强了Prior energy



Results



- Lazy Snapping 错误率仅是 Photoshop工具（Magnetic Lasso）的20%；
- 分割花费时间是Magnetic Lasso 的60%。





五. Bayesian Matting

Bayesian Matting（贝叶斯抠图）是利用统计前景与背景颜色的分布来计算alpha的值。Bayesian Matting需要由用户提供一张trimap，其中包括背景、前景和未知区域三部分。此算法的主要任务就是确立未知区域中每个像素对应的alpha的值。

大致思想

首先采集待计算的pixel周围的背景与前景颜色（包括已经计算出来的unknown区域的颜色）；

然后由采集的样本做出估计建立颜色分布的高斯模型；

最后，在贝叶斯框架下，通过MAP估计并利用迭代对F、B和alpha的各种可能进行计算，并取可能性最大的估计作为此pixel对应的结果。

Bayesian matting 从概率的角度考虑Matting问题，将图像的前景F，背景B和alpha值看做总概率分布，对于每一个待计算像素C,构造条件概率 $P(F, B, \alpha | C)$ ，根据概率论中的全概率公式，可得：

$$P(F, B, \alpha, C) = P(F, B, \alpha | C)P(C) = P(C | F, B, \alpha)P(F, B, \alpha)$$

再假设随机变量F,B和alpha分布独立，于是上式右边可进一步写成：

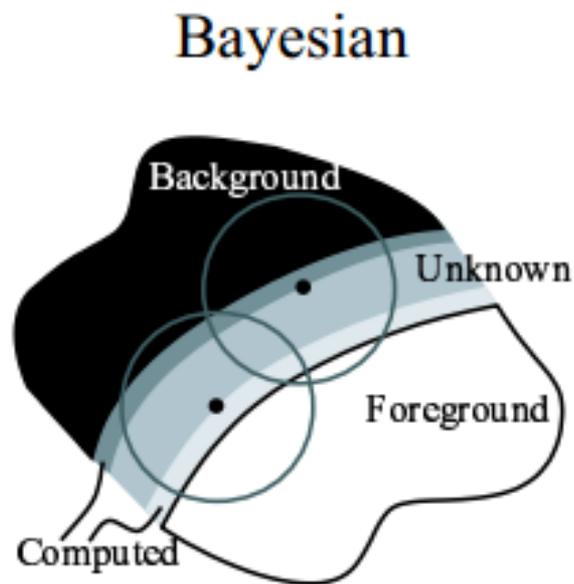
$$P(C | F, B, \alpha)P(F, B, \alpha) = P(C | F, B, \alpha)P(F)P(B)P(\alpha)$$

结合之前两式，可得到该问题的Bayesian等式：

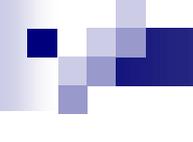
$$P(F, B, \alpha | C) = \frac{P(C | F, B, \alpha)P(F)P(B)P(\alpha)}{P(C)}$$

上式右端各项要么已知，要么可以通过邻域采样的方式进行估计，因而整个问题可以通过最大化后验概率（MAP）的方式进行求解。

Bayesian Matting的采样模型



算法采用了一个连续滑动窗口对邻域进行采样，窗口从未知区域和已知区域之间的两条边开始向内逐轮廓推进，计算过程也随之推进。

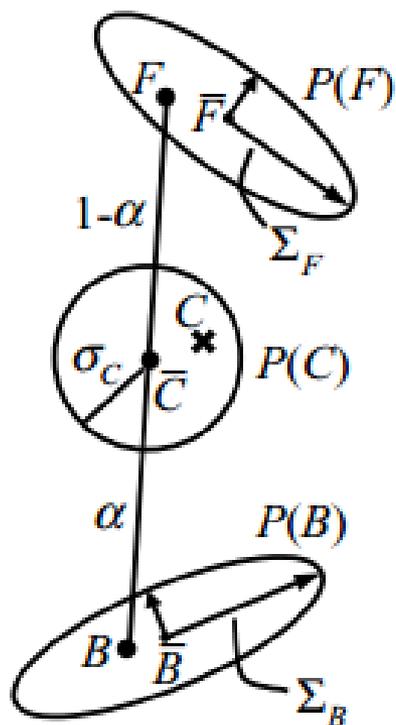


采样窗口定义为一个以待计算点为中心，半径 r 的圆域。采样窗口需要同时覆盖已知前背景区域和之前计算出的邻域像素点。

对采样点贡献度的加权原则：

1. 根据 α 值，前背景采样遵循越不透明的像素置信度越高；
2. 采样点到目标点之间的距离用一个高斯分布表示。

组合的权值被表示为:



$$\omega_i = \alpha^2 g_i \quad (\text{前景采样})$$

$$\omega_i = (1-\alpha)^2 g_i \quad (\text{背景采样})$$

Bayesian 框架

对于待计算像素C, 其服从的某一种颜色分布的概率最大化值可以等价的转化为若干个单独的Log似然函数的叠加

$$\begin{aligned} & \arg \max_{F, B, \alpha} P(F, B, \alpha | C) \\ &= \arg \max_{F, B, \alpha} P(C | F, B, \alpha) P(F) P(B) P(\alpha) / P(C) \\ &= \arg \max_{F, B, \alpha} L(C | F, B, \alpha) + L(F) + L(B) + L(\alpha) \end{aligned}$$

上式中 $L(\cdot)$ 表示Log似然函数, $L(\cdot) = \log P(\cdot)$, 这样做的目的是在等价的条件下将乘法转化为加法。

现在问题被转化为如何定义Log似然函数：

$L(C|F, B, \alpha)$, $L(F)$, $L(B)$ 和 $L(\alpha)$ 。算法的第一项为观察颜色 C 与预测颜色 C' 之间的误差，预测颜色可表示为：

$$L(C|F, B, \alpha) = -\|C - \alpha F - (1 - \alpha)B\|^2 / \delta_c^2$$

上式表示了一个均值为 $\bar{C} = \alpha F + (1 - \alpha)B$ ，标准差为 σ_C 的高斯概率分布的误差函数。

算法在颜色空间域一致性假设下，对 $L(F)$ 进行了估计。在获得所需的前背景采样以及对应的权值后，对采样值进行聚类。对每一个聚类，可计算出加权均值 \bar{F} 和加权协方差矩阵 Σ_F ：

$$\bar{F} = \frac{1}{W} \sum_{i \in N} w_i F_i$$

$$\Sigma_F = \frac{1}{W} \sum_{i \in N} w_i (F_i - \bar{F})(F_i - \bar{F})^T$$

其中

$$W = \sum_{i \in N} w_i$$

前景似然函数 $L(F)$ 可建模为一个有向高斯分布:

$$L(F) = -\frac{1}{2} (F - \bar{F})^T \Sigma_F^{-1} (F - \bar{F})$$

Bayesian Matting求解过程

在转化为若干个单独的Log似然函数叠加后，Bayesian matting方法将求解问题分为两个子问题进行计算。

在第一步，假设alpha值为一个常数，分别关于F和B求偏导，并令其值为0，得到：

$$\begin{bmatrix} \Sigma_F^{-1} + I\alpha^2/\sigma_C^2 & I\alpha(1-\alpha)/\sigma_C^2 \\ I\alpha(1-\alpha)/\sigma_C^2 & \Sigma_B^{-1} + I(1-\alpha)^2/\sigma_C^2 \end{bmatrix} \begin{bmatrix} F \\ B \end{bmatrix} \\ = \begin{bmatrix} \Sigma_F^{-1}\bar{F} + C\alpha/\sigma_C^2 \\ \Sigma_B^{-1}\bar{B} + C(1-\alpha)/\sigma_C^2 \end{bmatrix},$$

其中 I 是一个 3×3 的单位矩阵，我们通过求解一个 6×6 的线性方程组得到最佳的估计色 F 和 B 。

第二步，假设 F 和 B 是常数，从而得到关于 α 的二次方程，并关于 α 求导，令其值为 0，得到：

$$\alpha = \frac{(C - B) \cdot (F - B)}{\|F - B\|^2}$$

- 优化估计通过反复迭代上述第一步和第二步完成，首先用待计算像素周围的 α 值的平均值作为该点第一次迭代的 α 值，之后循环重复第一步和第二步，直到 α 值的变化足够小或者迭代次数高于某一个阈值时停止。
- 当有多个前背景聚类时，分别对每一对前背景聚类分别优化求解，最后根据后验概率值大小决定作为最终计算结果的一组解。

Results

Input



Segmentation

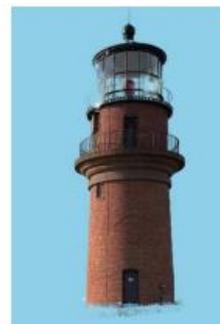


Composite

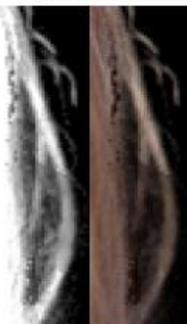


不同方法对比

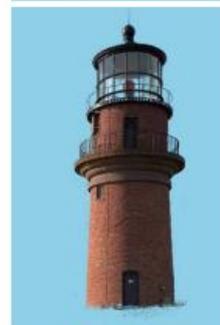
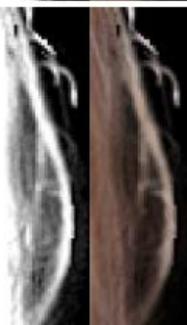
Knockout



Ruzon and Tomasi



Bayesian approach



Alpha Matte

Composite

Inset

Alpha Matte

Composite

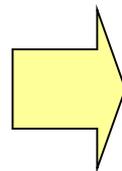
Inset

六.Closed Form Matting

抠像 (matting)&融合



+

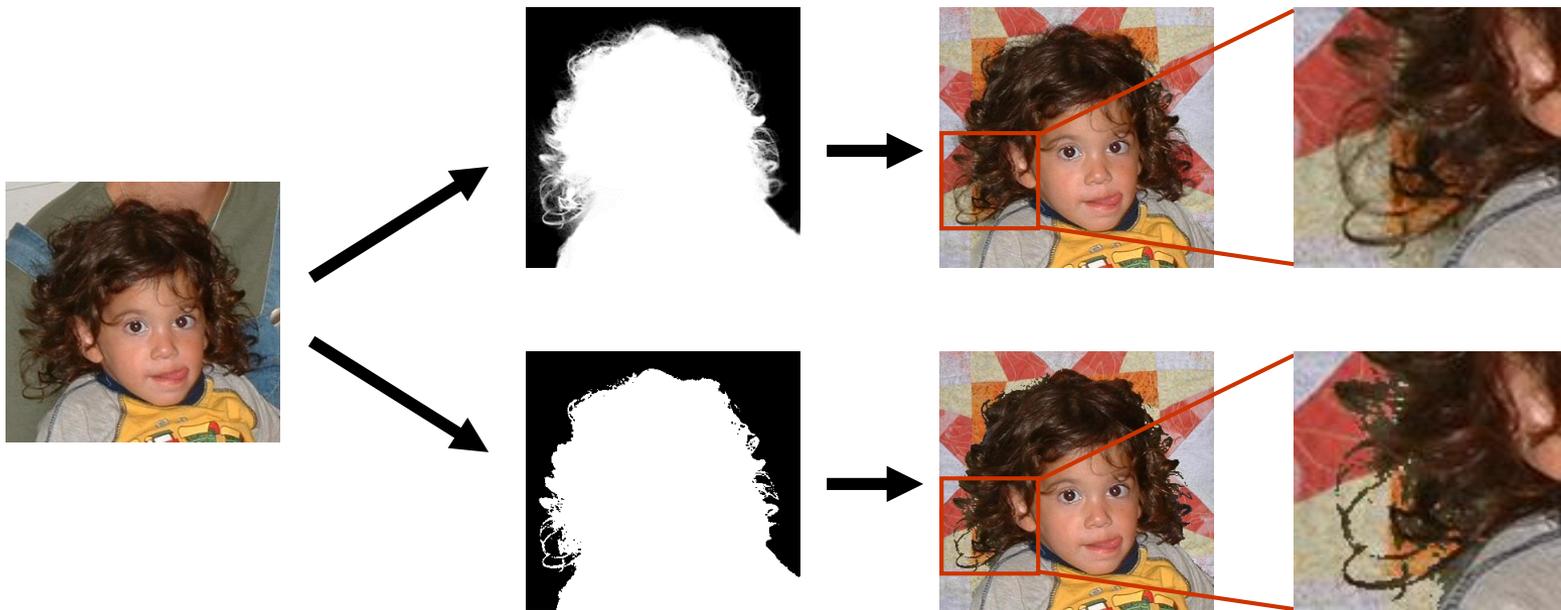


matting 公式

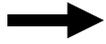
$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i$$

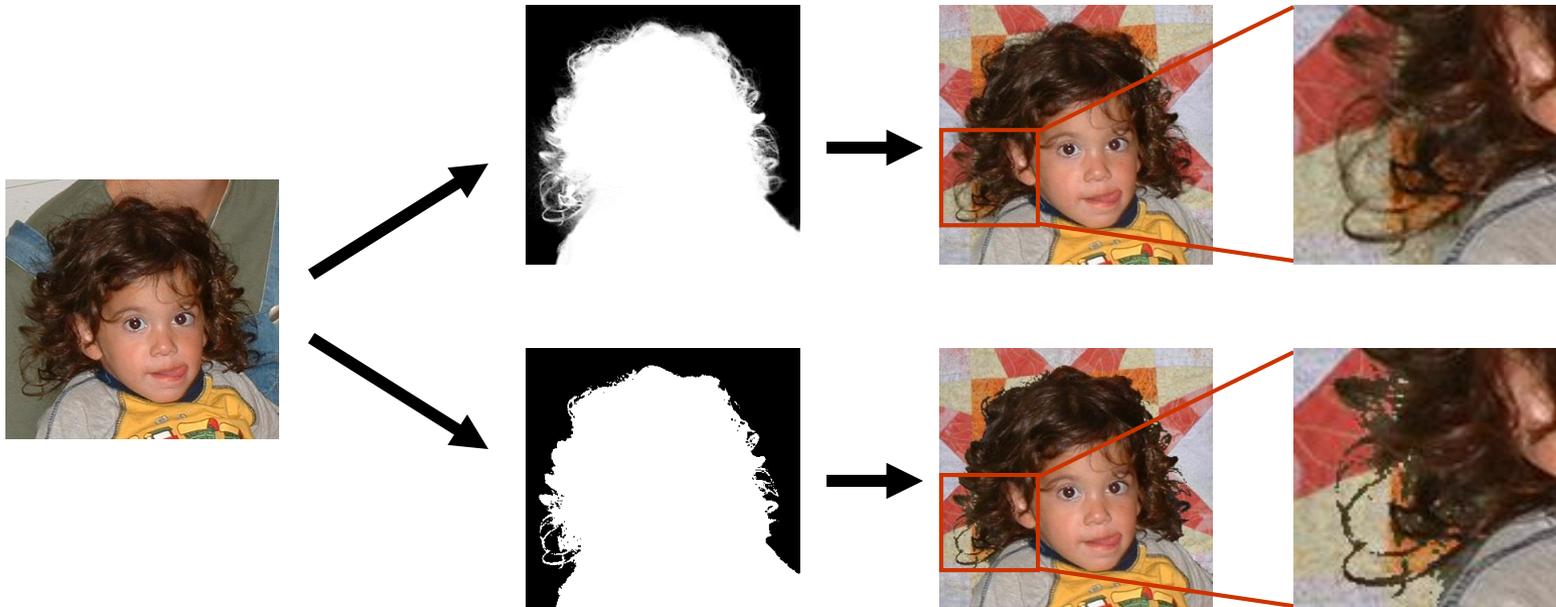


matting 难在？

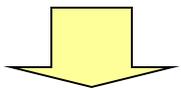






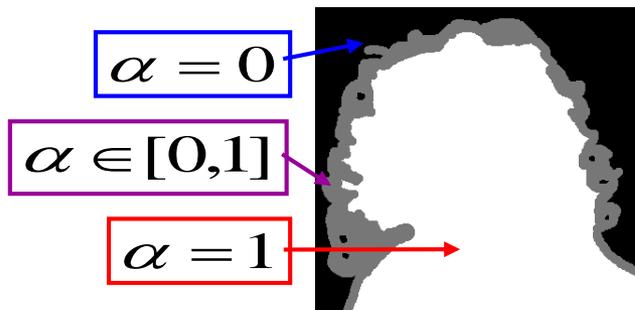


$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i$$



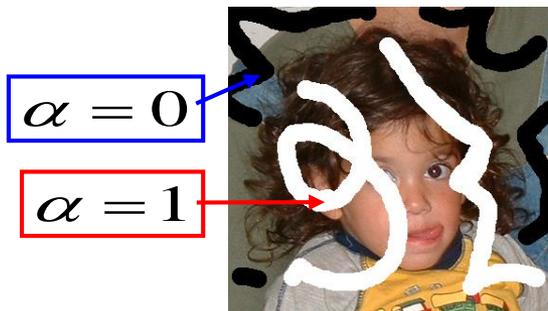
Matting 是个 ill posed 问题: 7 个未知量, 但 每个 pixel 仅有 3 个约束。

之前的研究方法



trimap :

- Bayesian Matting (Chuang et al, CVPR01)
- Poisson Matting (Sun et al SIGGRAPH 04)
- Random Walk (Grady et al 05)



用户交互界面:

- Wang&Cohen ICCV05

基于 trimap 方法存在的问题

- 迭代的求解 F, B 和 α
- 需要准确的 trimap

Input Scribbles



Bayesian matting from scribbles



Good matting from scribbles



(Replotted from Wang&Cohen)

scribbles 方法 (Wang&Cohen ICCV05)



$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i$$

- 迭代地求解 F, B 和 α
- 每次迭代都涉及复杂的非线性优化

Closed Form

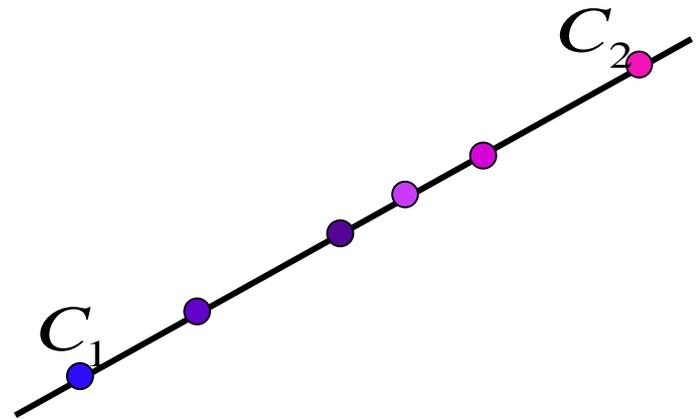
$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i$$

- 解析消除 **F,B**，以二次代价获得 α ；
- 可证的正确的结果；
- 结果的定量评价。

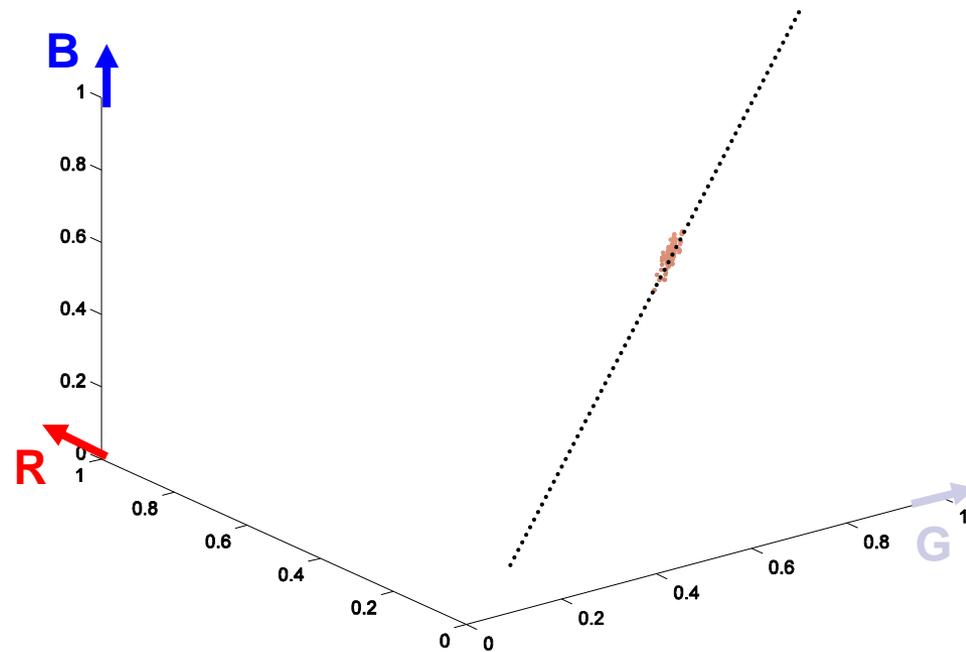
颜色线 (Color lines)

Color Line: $\{C_i \in R^3 \mid C_i = \beta_i C_1 + (1 - \beta_i) C_2\}$

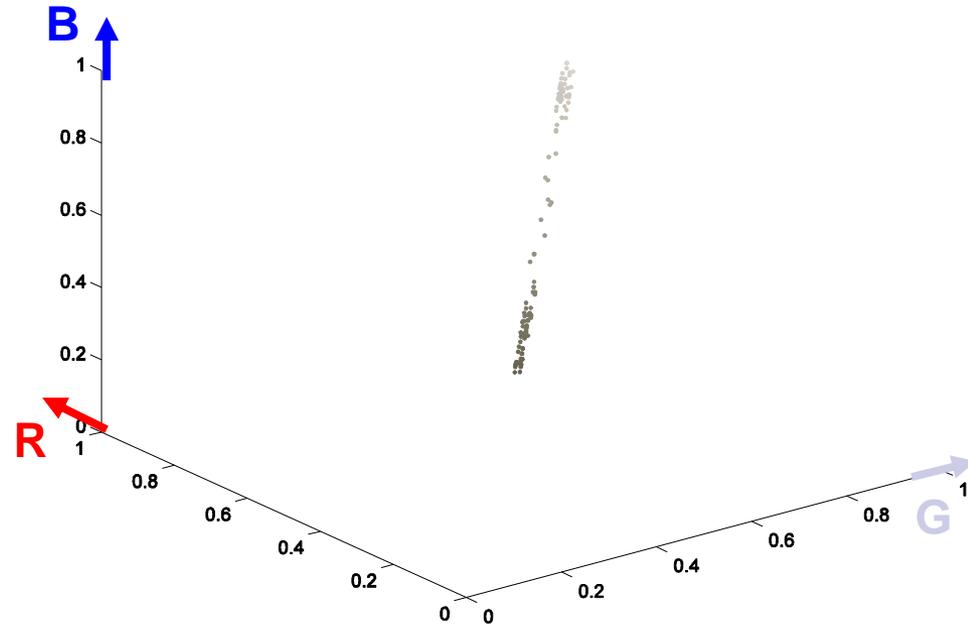
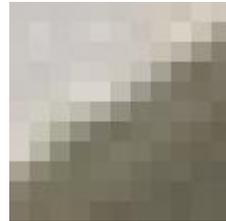
(Omer&Werman 04)



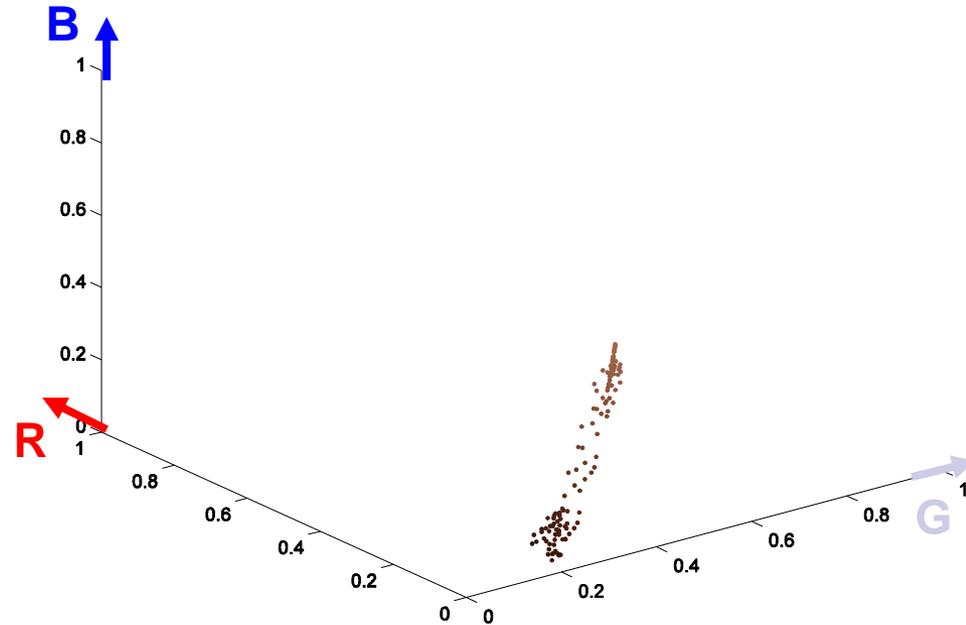
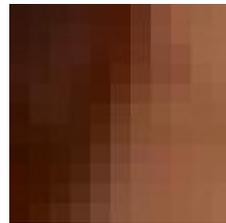
Color Line: $\{C_i \in R^3 \mid C_i = \beta_i C_1 + (1 - \beta_i) C_2\}$



Color Line: $\{C_i \in R^3 \mid C_i = \beta_i C_1 + (1 - \beta_i) C_2\}$



Color Line: $\{C_i \in R^3 \mid C_i = \beta_i C_1 + (1 - \beta_i) C_2\}$



根据color lines得到的线性模型

如果在局部窗口中，前景 **F**,背景**B** 的像素颜色落在**color lines**上，则有：

$$\alpha_i = a^R \color{red}{\mathbf{I}_i} + a^G \color{green}{\mathbf{C}_i} + a^B \color{blue}{\mathbf{I}_i} + b \quad \forall i \in w$$

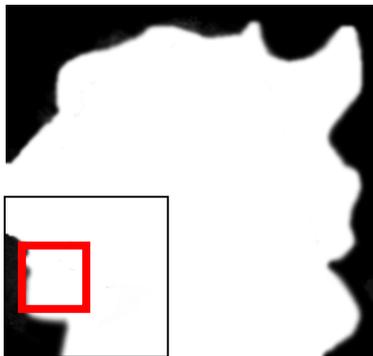


$$\text{Mask} = -2 \cdot \text{Original} + 1$$

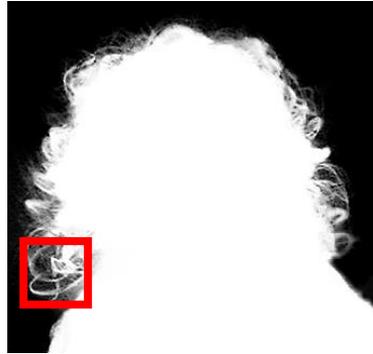
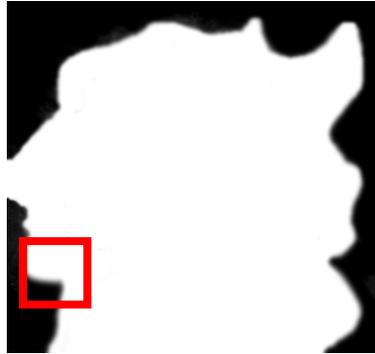
➡ 因此：前景**F**,背景**B** 能够从matting cost消除

估计 α -matte

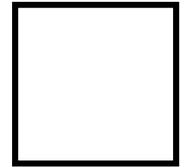
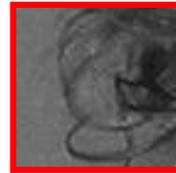
? ...



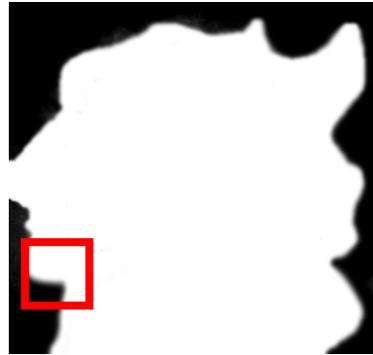
? ...



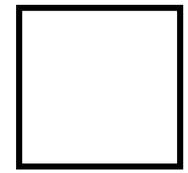
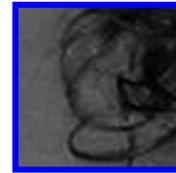
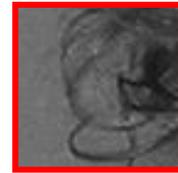
...?→



? ...



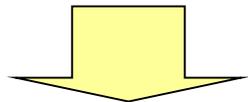
...?...



$$J(\alpha) = \sum_{w \in I} d(\alpha_w, \text{Span}\{\mathbf{R}_w, \mathbf{G}_w, \mathbf{B}_w, 1\}) + \varepsilon \cdot \text{smoothness}(\alpha)$$

计算定理

前景**F**,背景**B** 局部地落在 **color lines** 上



$$\begin{aligned} J(\alpha) &= \sum_{w \in I} d(\alpha_w, \text{Span}\{\mathbf{R}_w, \mathbf{G}_w, \mathbf{B}_w, \mathbf{1}\}) \\ &= \alpha^T L \alpha \end{aligned}$$

这里, $L(i, j)$ 是图像的局部函数

$$L(i, j) \propto \sum_{k|(i, j) \in w_k} -\left(1 + (\mathbf{C}_i - \mu_k)^T (\Sigma_k + \varepsilon \mathbf{I}_3)^{-1} (\mathbf{C}_j - \mu_k)\right)$$

应用线性代数求解 α

输入:

图像+ 用户交互

$$\alpha = \arg \min \alpha^T L \alpha$$

$$s.t. \quad \alpha_i = 0, \quad i \in \text{mask}$$

$$\alpha_i = 1, \quad i \in \text{contour}$$



$$L(i, j) \propto \sum_{k|(i,j) \in w_k} -\left(1 + (C_i - \mu_k)^T (\Sigma_k + \epsilon I_3)^{-1} (C_j - \mu_k)\right)$$

最小化代价和解决方案

定理:

Given: $I = \alpha^* F^* + (1 - \alpha^*) B^*$

If:

• F^*, B^* locally on color lines

• Constraints consistent with α^*

Then:

$$\alpha^* = \arg \min \alpha^T L \alpha$$

$$s.t. \quad \alpha_i = 0, \quad i \in \text{black shape}$$

$$\alpha_i = 1, \quad i \in \text{white shape}$$

Matting 和 spectral 分割

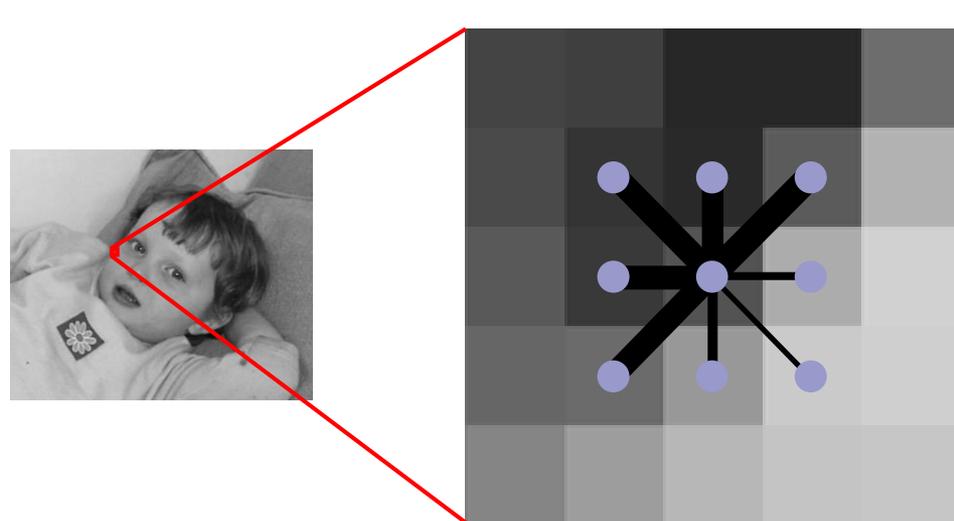
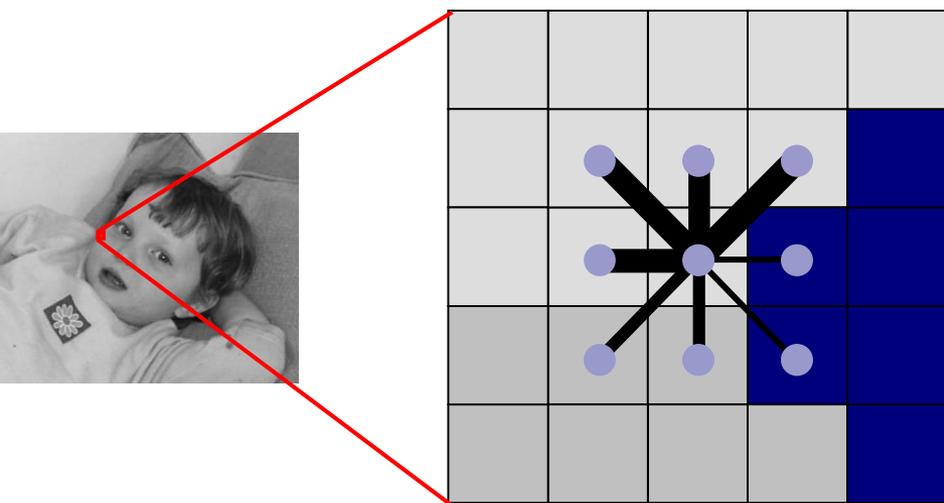
Spectral 分割: 分析拉普拉斯图L的最小特征向量

$$L = D - W$$

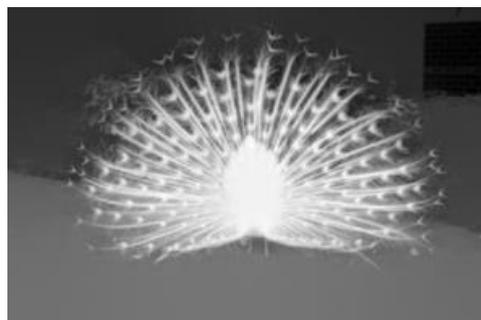
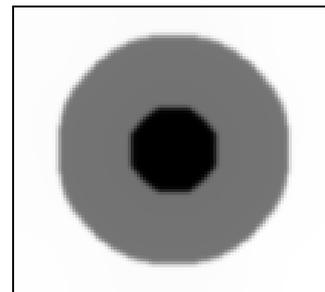
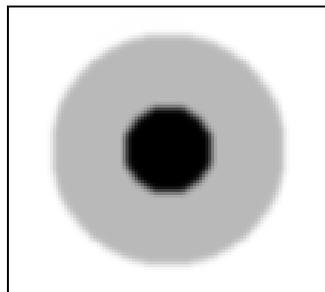
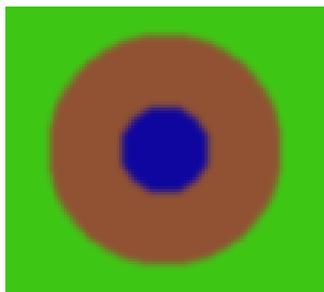
$$D(i, i) = \sum_j W(i, j)$$

$$W_{Global}(i, j) = e^{-\|C_i - C_j\|^2 / \sigma^2}$$

$$W_{Matting}(i, j) \propto \sum_{k|(i, j) \in w_k} \left(1 + (C_i - \mu_k)^T (\Sigma_k + \epsilon I_3)^{-1} (C_j - \mu_k) \right)$$



特征向量比较



Input image

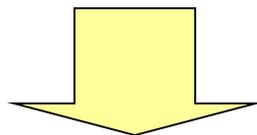
**Matting
Eigenvectors**

**Global- σ
Eigenvectors**

Matting 结果



+



更多链接:

1. 斯坦福公开课, 高斯混合模型:

http://v.163.com/movie/2008/1/L/3/M6SGF6VB4_M6SGKK6L3.html

2. 维基百科:

http://en.wikipedia.org/wiki/Mixture_model#Gaussian_mixture_model

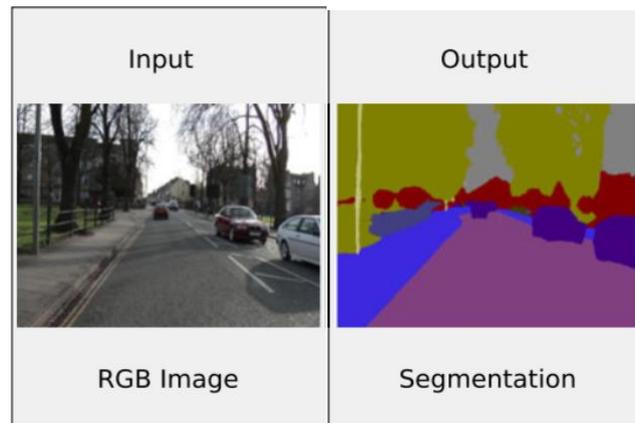
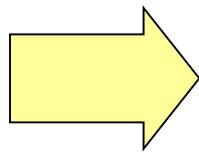
3. Alpha Matting Evaluation Website

<http://www.alphamatting.com/>

七、基于深度学习的图像分割



普通分割



语义分割

FCN: Fully Convolutional Networks

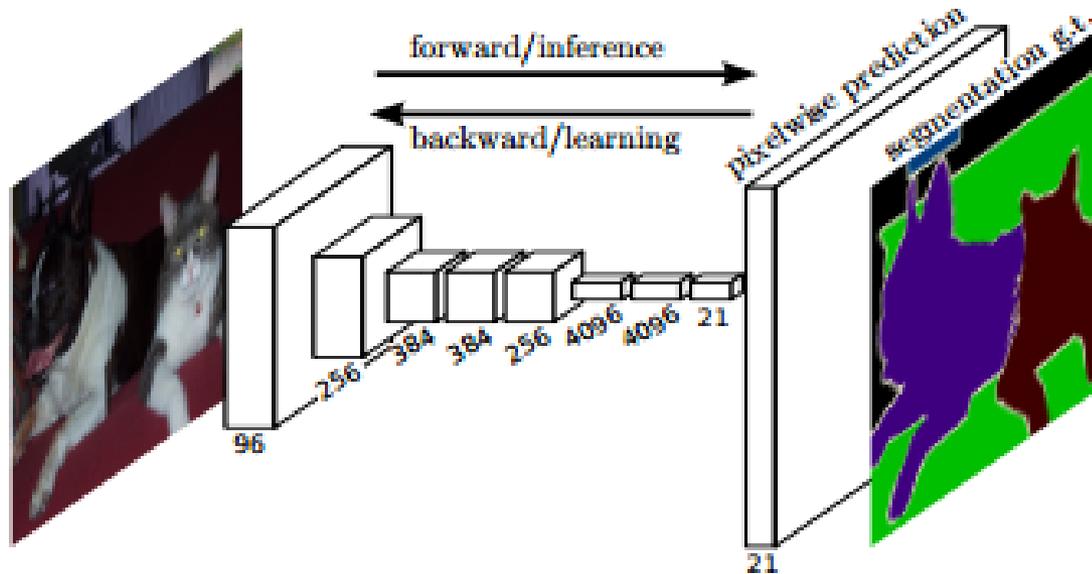
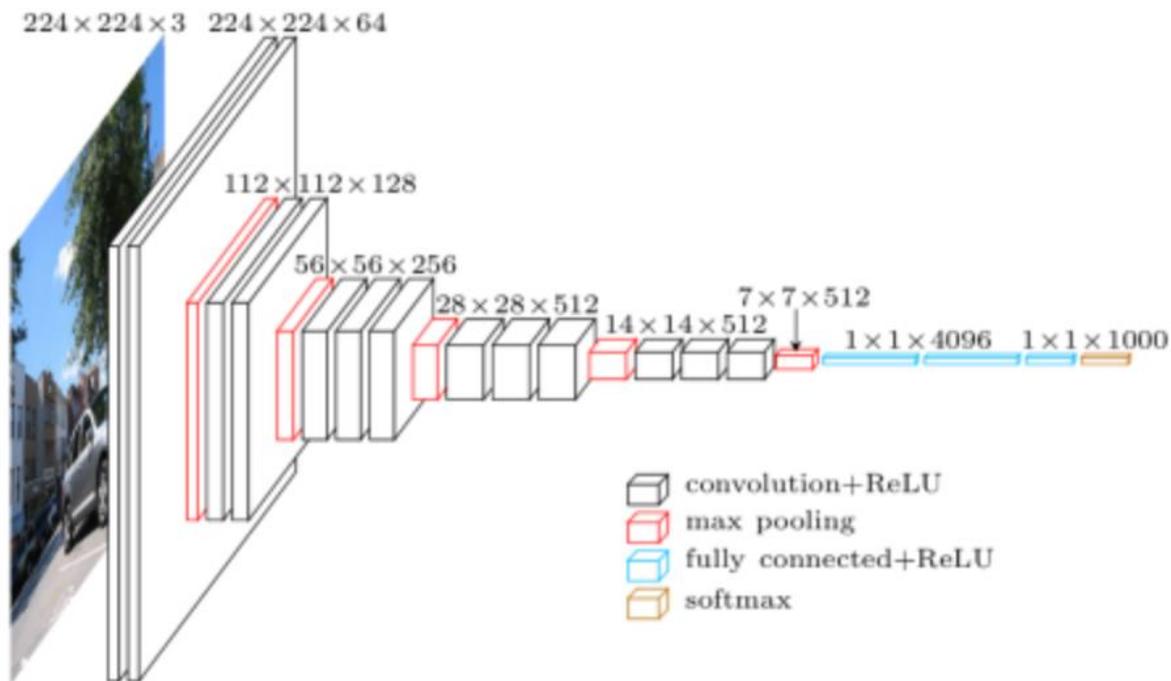


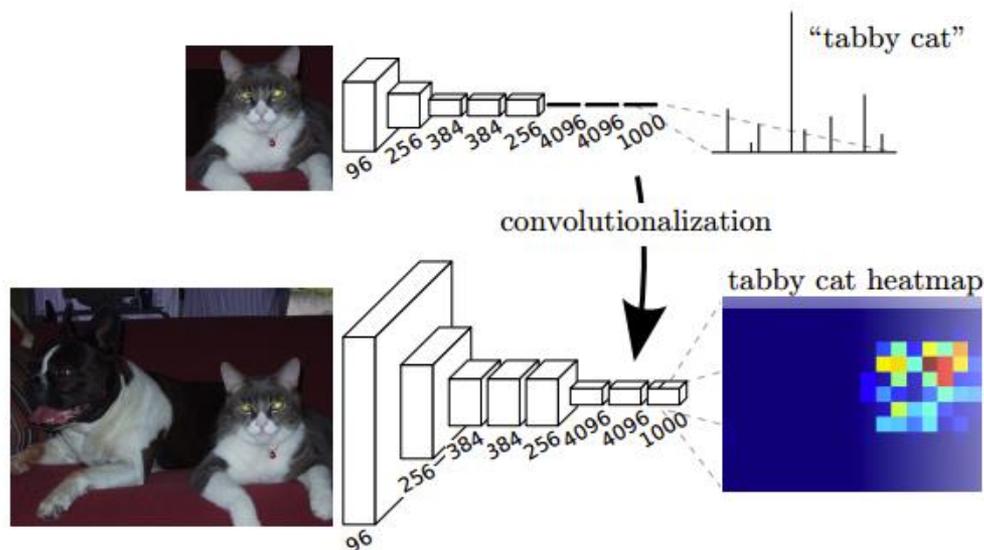
Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

FCN的参考：VGG-Nets

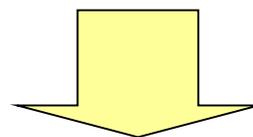


2014年ImageNet竞赛定位任务的第一名和分类任务的第二名的中的基础网络。

FCN的实现



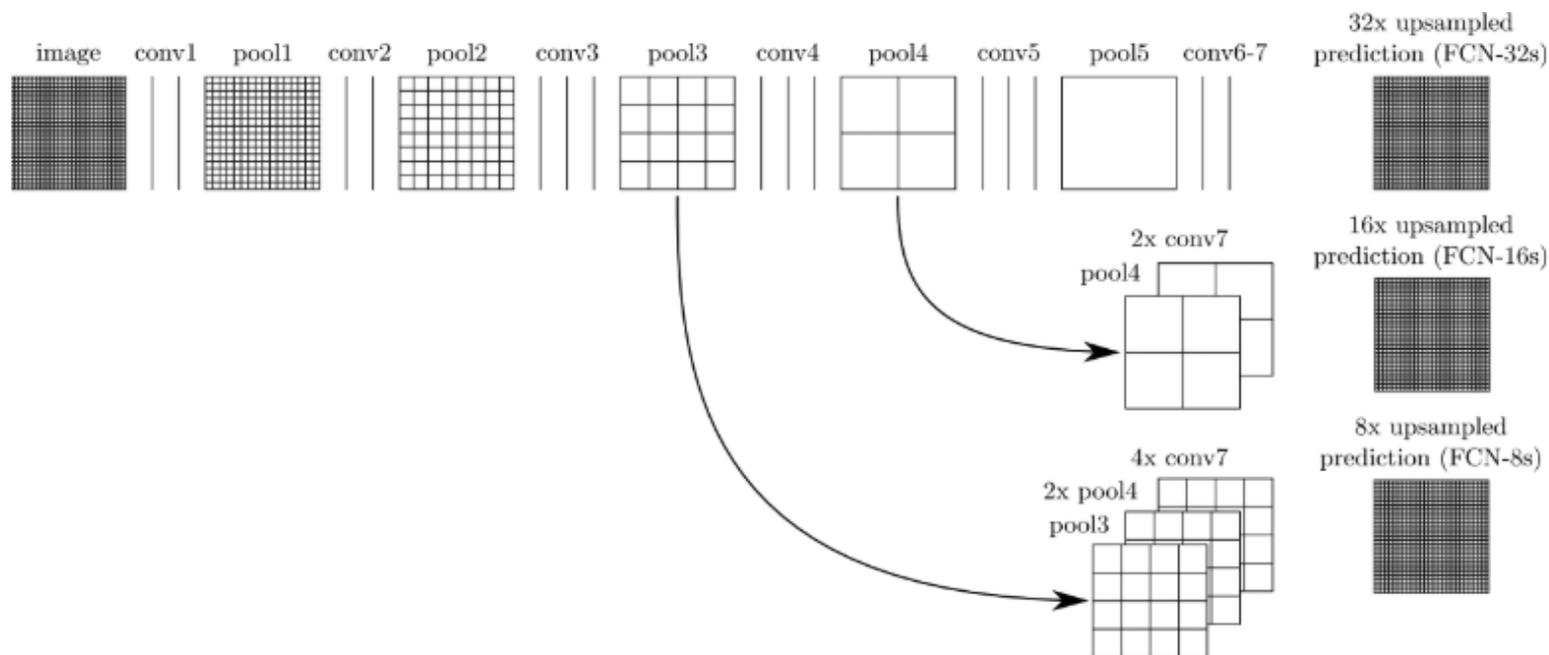
全连接层



卷积层

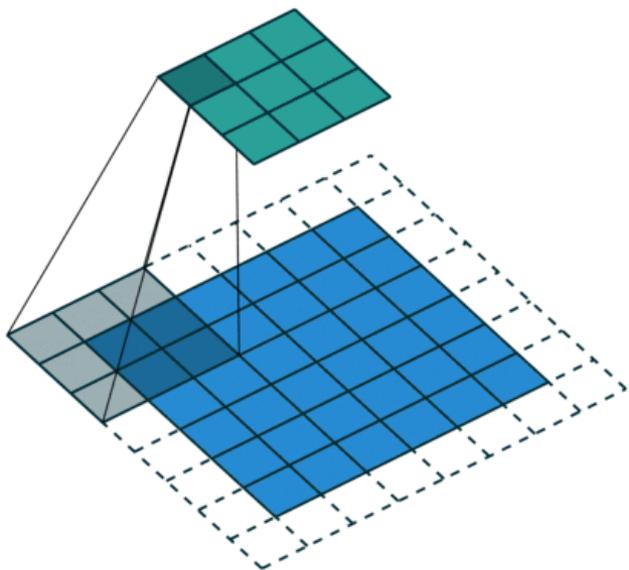
Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

FCN的实现

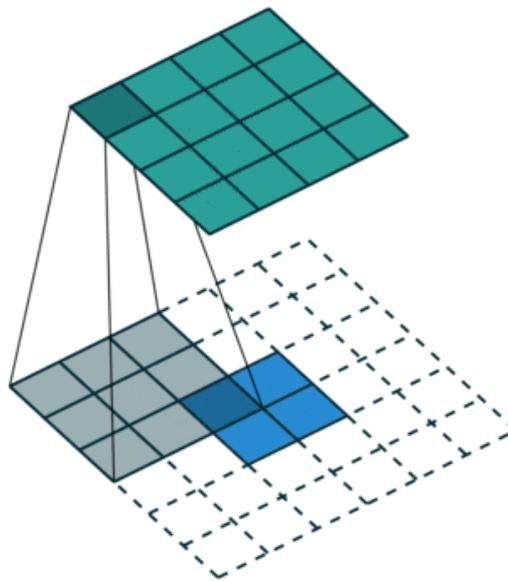


- FCN-32S:对conv 7的特征图进行步长为32的上采样,还原到原始图片大小。
- FCN-16S:对conv 7的特征图进行步长为 2的上采样后与pool4层进行特征融合。
- FCN-8S:同上, 进行三个层次间的上采样, 还原到原始图片大小。

上采样:Deconvolution



卷积



反卷积（转置卷积）

FCN的优劣势

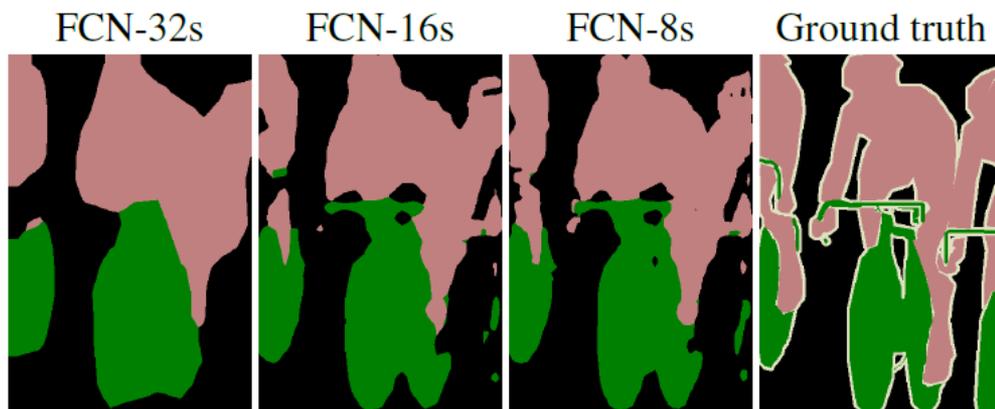
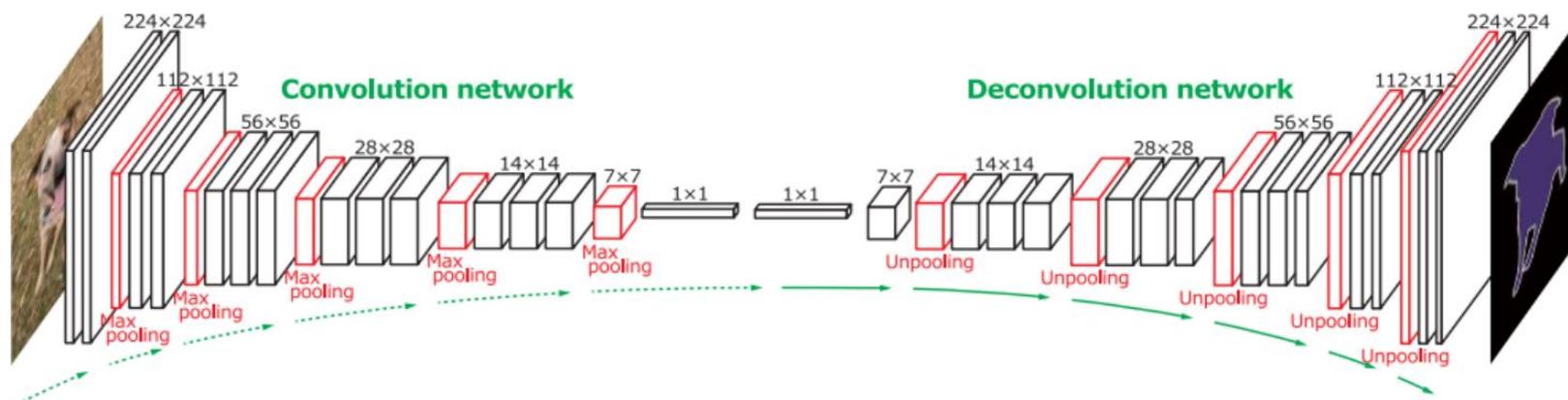


Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

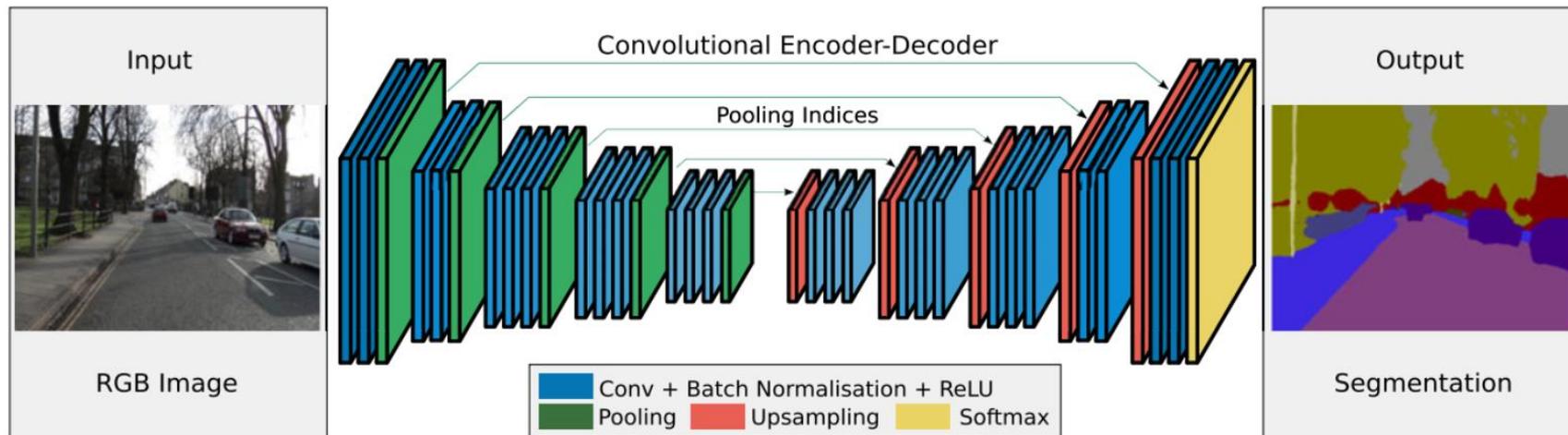
- 优势：
实现简单。
- 劣势：
 1. 实现效果比较粗糙。
 2. 反卷积层训练困难。
 3. 网络结构不够美观。

DeconvNet



- Encoder-Decoder结构
- Encoder末尾为2个全连接层

SegNet



- 在卷积层加入Batch Normalisation
- 没有全连接层

PSPNet: Pyramid Scene Parsing Network

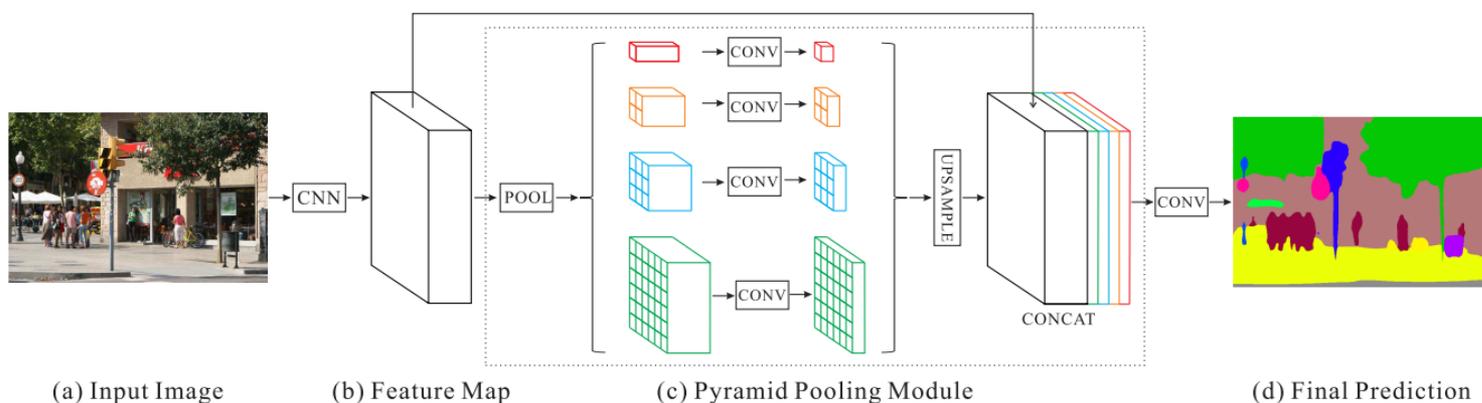
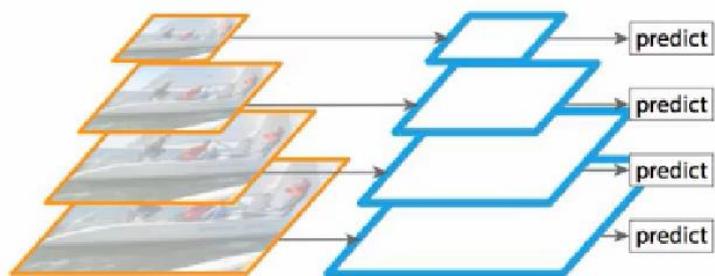


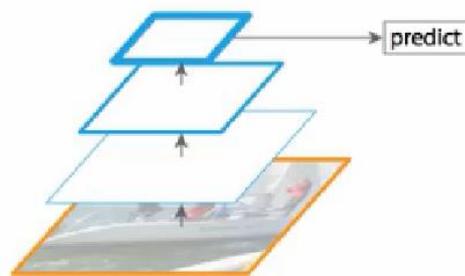
Figure 3. Overview of our proposed PSPNet. Given an input image (a), we first use CNN to get the feature map of the last convolutional layer (b), then a pyramid parsing module is applied to harvest different sub-region representations, followed by upsampling and concatenation layers to form the final feature representation, which carries both local and global context information in (c). Finally, the representation is fed into a convolution layer to get the final per-pixel prediction (d).

- 加入全局金字塔池化模块

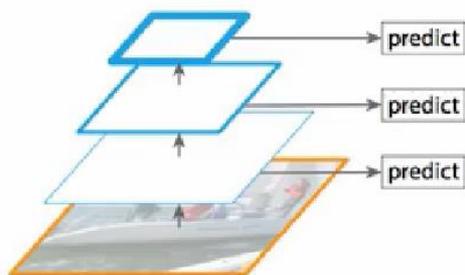
FPN:特征金字塔网络



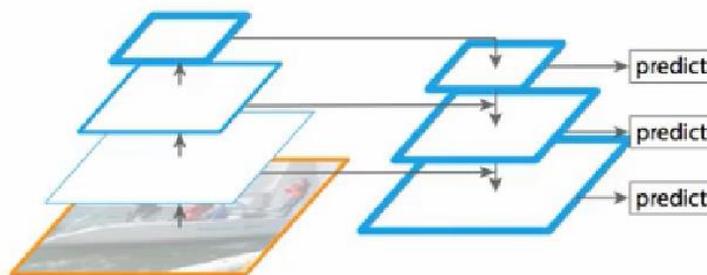
(a) Featurized image pyramid



(b) Single feature map



(c) Pyramidal feature hierarchy



(d) Feature Pyramid Network

Mask-RCNN

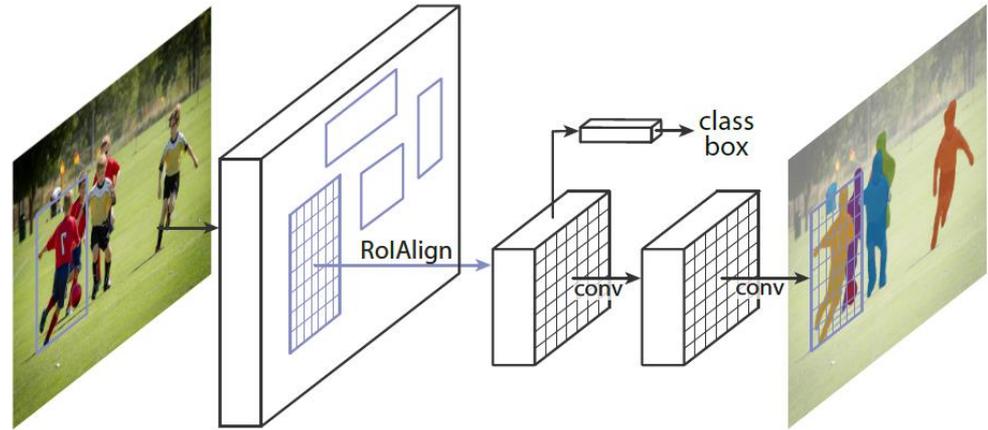
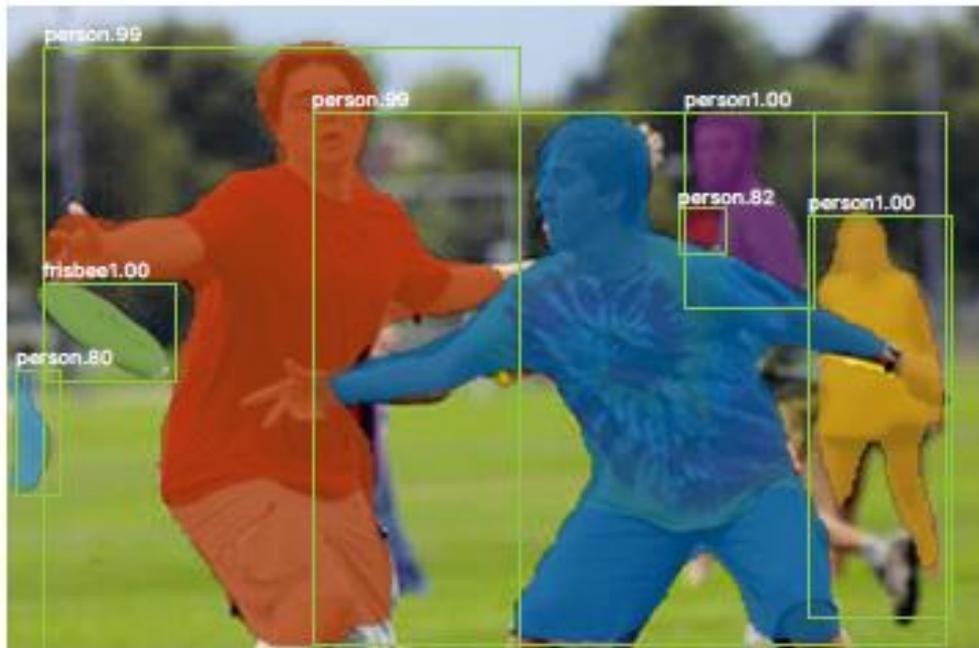


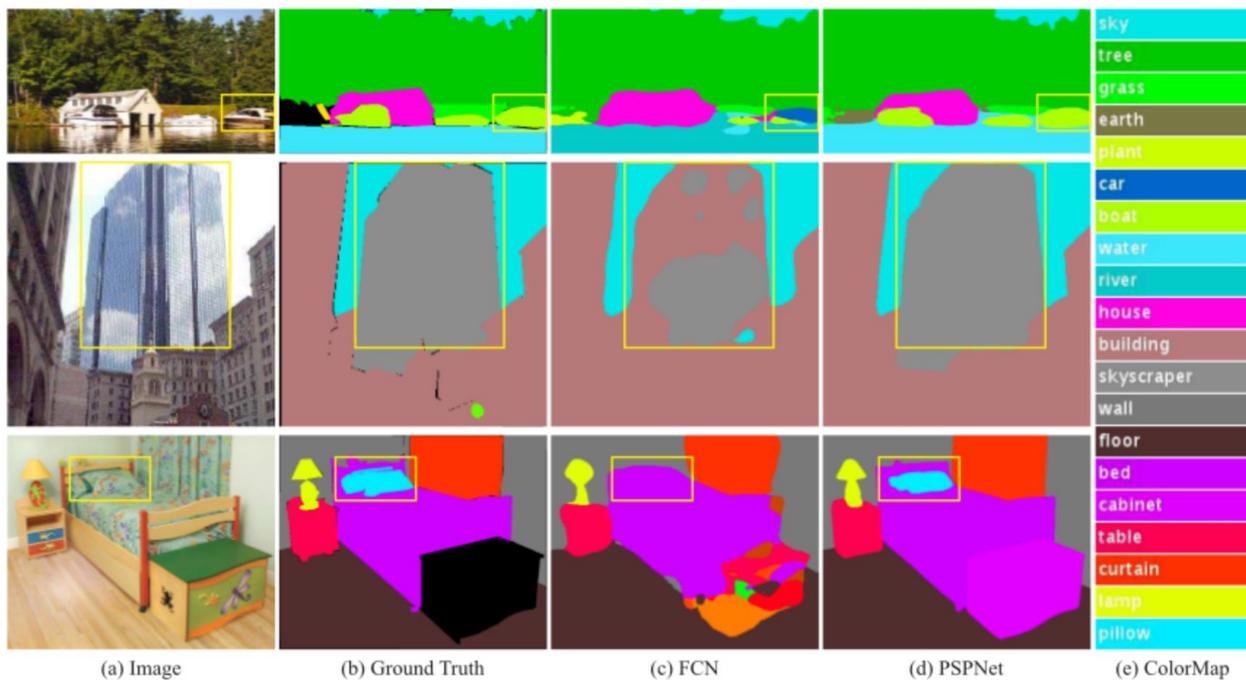
Figure 1. The **Mask R-CNN** framework for instance segmentation.



Mask-RCNN,

Kaiming He,Georgia Gkioxari,Piotr Doll'ar,Ross Girshick,arxiv 1703.06870

基于深度学习的图像分割



框架总结:

1. 整体网络由一个下采样过程与一个上采样过程搭建。
(卷积、反卷积)
2. 网络使用多尺度的特征进行融合。(特征金字塔网络)
3. 最终获得像素级别的图像分割: 对每一个像素点进行判断类别。

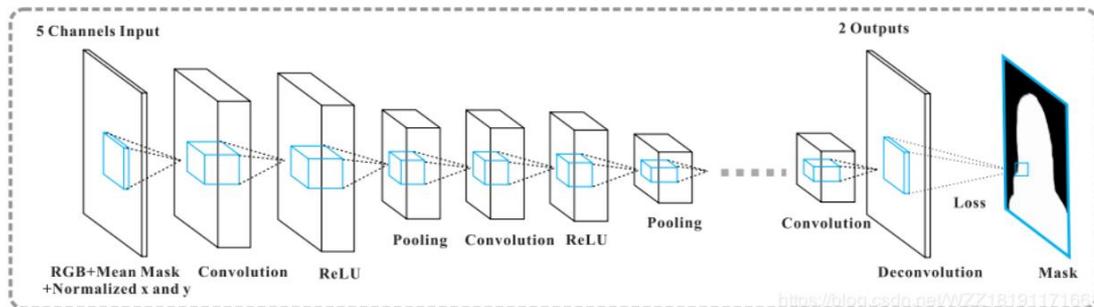
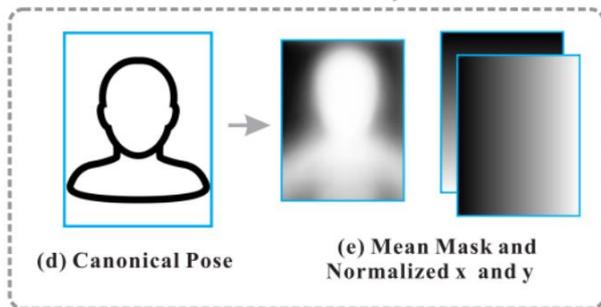
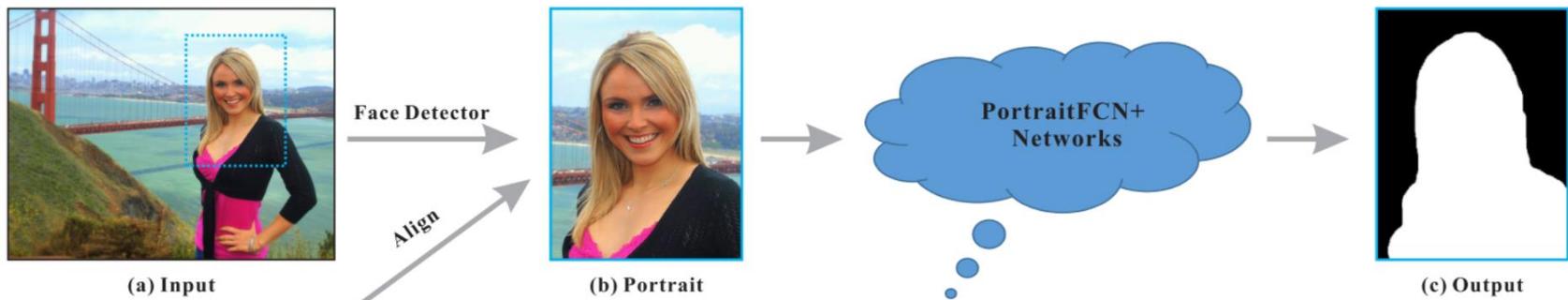
八、基于深度学习的抠像



- 第一篇具有较好效果的人像抠图文章

Automatic Portrait Segmentation for Image Stylization,
Xiaoyong Shen Aaron Hertzmann Jiaya Jia Sylvain Paris Brian Price Eli Shechtman Ian Sachs

PortraitFCN+



计算平均模板与对齐

$$M = \frac{\sum_i m_i \cdot T_i(M^i)}{\sum_i m_i},$$

对于每个肖像-掩码对 $\{P_i, M_i\}$,使用单映矩阵 T_i 将其变换。其中 m_i 是一个矩阵,若 M_i 中的像素在经过 T_i 变换后在图像外,将其置为1,否则将其置为0,使用这种方式获得平均模板 M 。

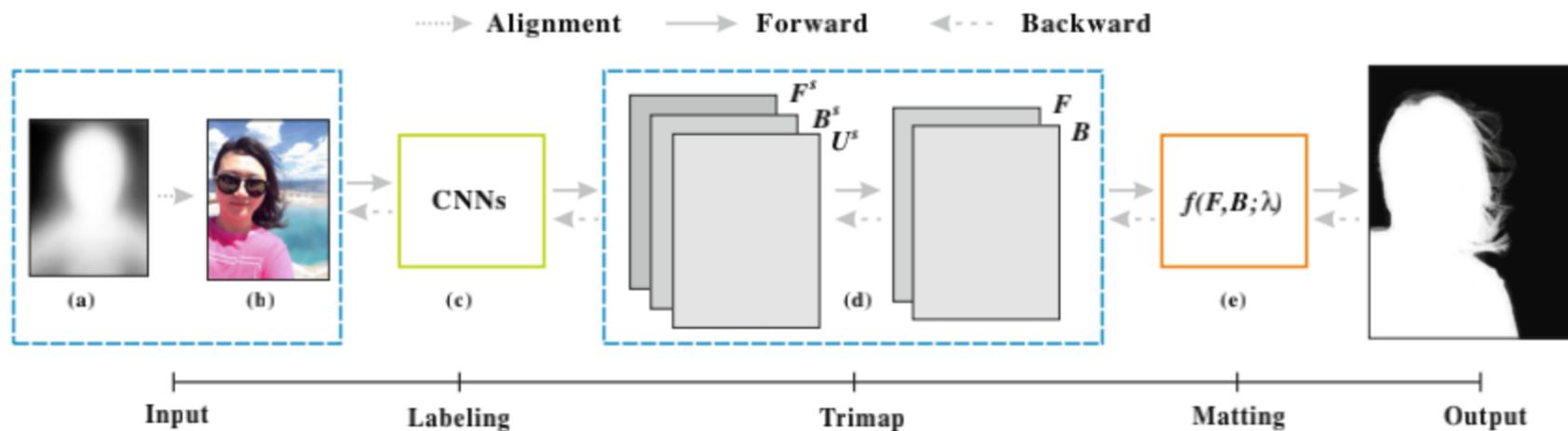
PortraitFCN+算法的突出点



- 添加肖像位置通道 x 、 y 和形状通道 m 到网络中。
- 让网络更加关注于人脸周围的区域。

Figure 5: *Some example portrait images with different variations in our dataset.*

Deep Automatic Potrait Matting



在PotraitFCN+的基础上获得了：

- 更丰富的抠图细节。
- 确切的 α -matte。

Image Matting Layer

$$F = \frac{\exp(F^s)}{\exp(F^s) + \exp(B^s) + \exp(U^s)}$$

Trimap Labeling的输出是F、B、U层的概率。根据输出，使用softmax公式得到前景F和背景B的概率图。

Image Matting Layer

之后使用以下公式估计 α -matte。

$$\min \lambda \mathbf{A}^T \mathbf{B} \mathbf{A} + \lambda (\mathbf{A} - \mathbf{1})^T \mathbf{F} (\mathbf{A} - \mathbf{1}) + \mathbf{A}^T \mathcal{L} \mathbf{A},$$

为了求得 \mathbf{A} ，将Image Matting Layer层的函数这样设计：

$$f(F, B; \lambda) = \lambda(\lambda \mathbf{B} + \lambda \mathbf{F} + \mathcal{L})^{-1} F,$$

α -matte损失函数

论文中没有采用一般的L1，L2范式作为损失函数。这是因为label大多数是0和1。如果使用这两个范式会不利于学习。因此论文考虑对每一个 α 值进行加权得到最终的损失函数。

$$L(\mathcal{A}, \mathcal{A}^{gt}) = \sum_i w(\mathcal{A}_i^{gt}) \|\mathcal{A}_i - \mathcal{A}_i^{gt}\|,$$

$$w(\mathcal{A}_i^{gt}) = -\log(p(A = \mathcal{A}_i^{gt})),$$

论文实现的效果



(a) Input

(b) Strokes

(c) Trimap

(d) AutoTrimap [3]



(e) Ours

(f) Matte of (b)

(g) Matte of (c)

(h) Matte of (d)

Deep Image Matting

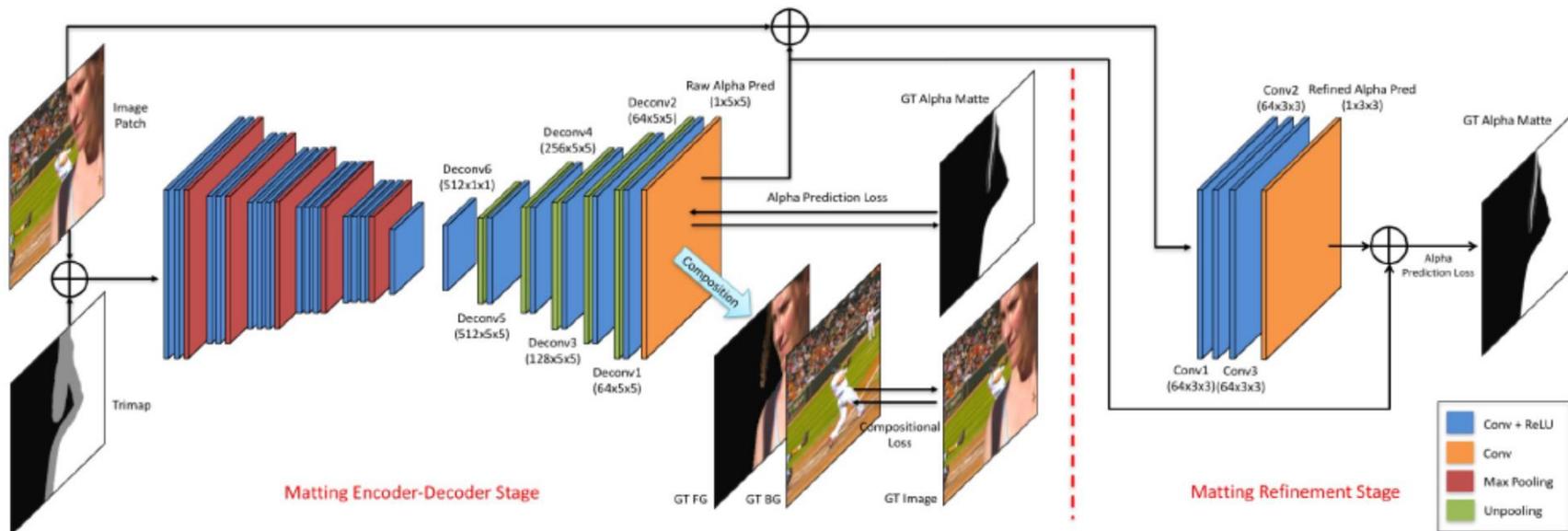


Figure 3. Our network consists of two stages, an encoder-decoder stage (Sec. 4.1) and a refinement stage (Sec. 4.2)

2017年在抠图alphamattng.com 比赛中，排名第一

第一部分：Matting encoder-decoder stage

使用Encoder-Decoder结构，对预测得到的 α -matte、图像进行估计。

其损失函数为：

$$\mathcal{L}_\alpha^i = \sqrt{(\alpha_p^i - \alpha_g^i)^2 + \epsilon^2}, \quad \alpha_p^i, \alpha_g^i \in [0, 1].$$

$$\mathcal{L}_c^i = \sqrt{(c_p^i - c_g^i)^2 + \epsilon^2}.$$

$$\mathcal{L}_{overall} = w_l \cdot \mathcal{L}_\alpha + (1 - w_l) \cdot \mathcal{L}_c;$$

其中， α 为 α -matte的像素值，而 c 为图像上点的像素值。

第二部分：Matting refinement stage

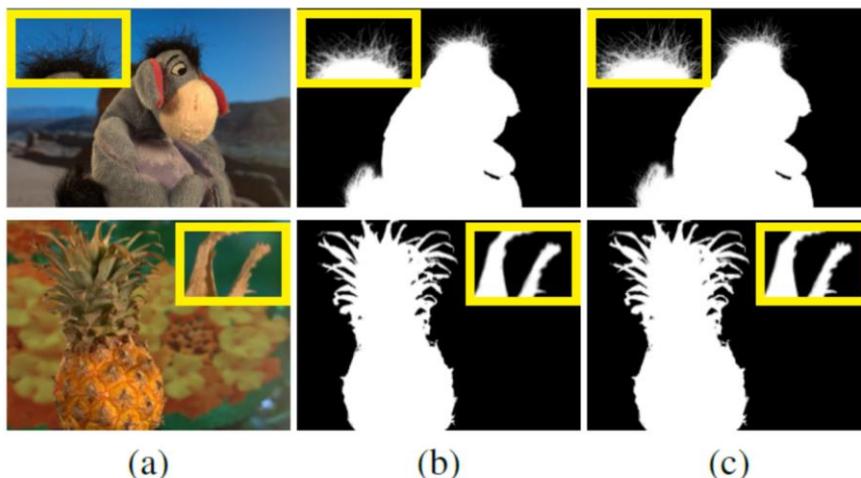
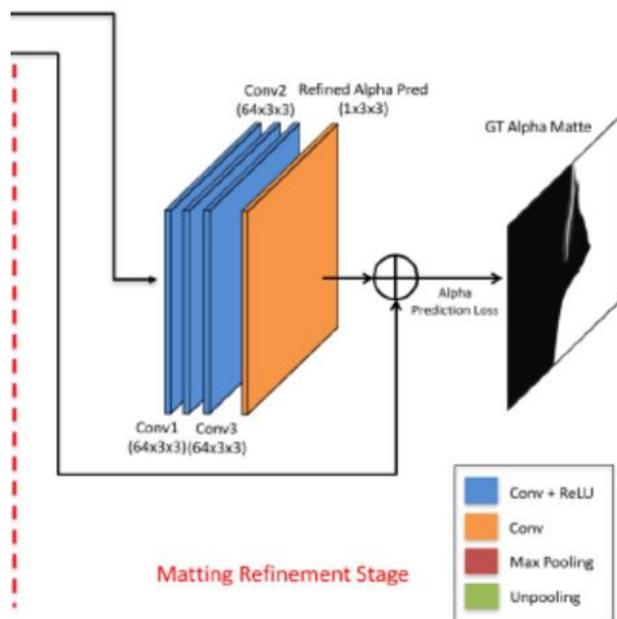


Figure 4. The effect of our matting refinement network. (a) The input images. (b) The results of our matting encoder-decoder stage. (c) The results of our matting refinement stage.

- 使用上图网络，锐化轮廓

与其它方法的对比效果图

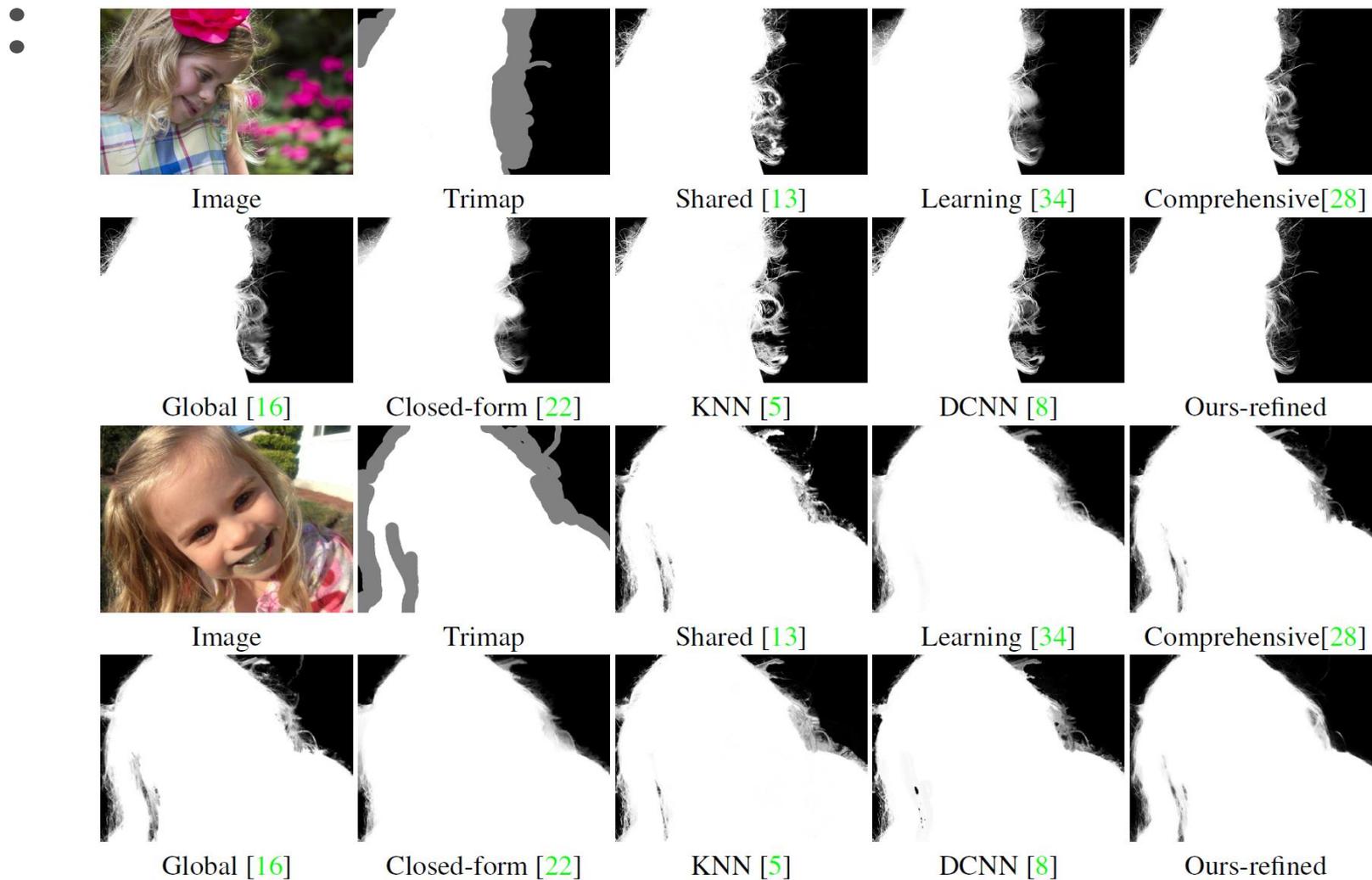


Figure 9. Example results from our real image dataset.

更多链接:

1. Detectron 开源代码:

<https://github.com/facebookresearch/Detectron>

2. PSPNet 测试代码

<https://github.com/hszhao/PSPNet>

3. Deep-image-matting 代码:

<https://github.com/Joker316701882/Deep-Image-Matting>



Thank you!