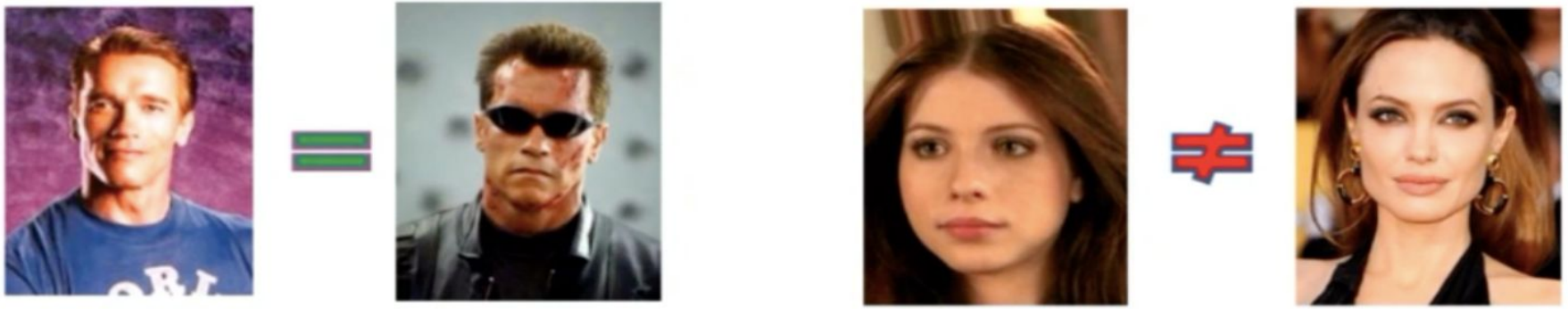
Metric Learning

на примере задачи Face Recognition

Георгий Сурин

Задача Face Recognition

Верификация (1:1)

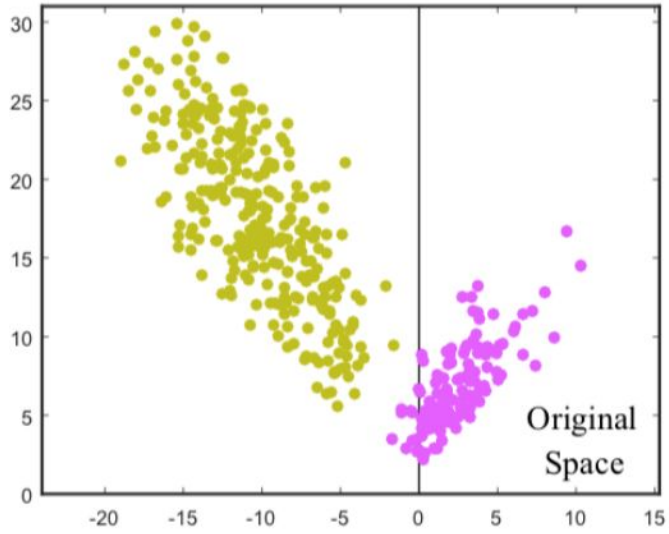


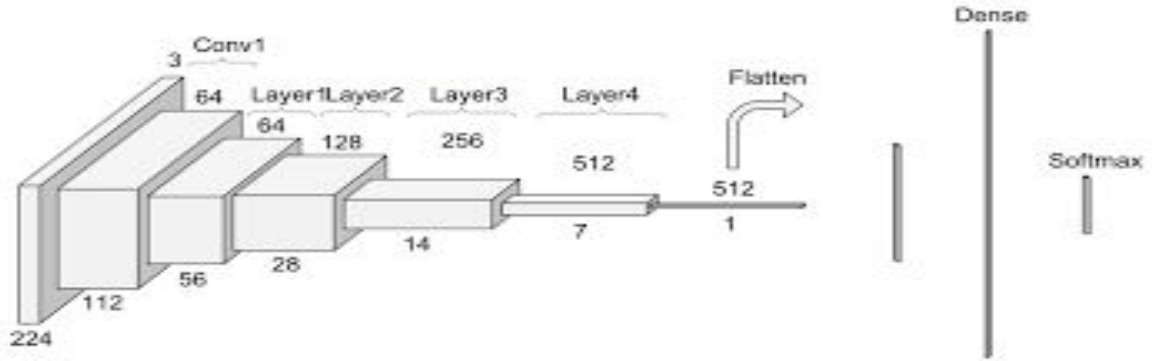
Идентификация (1:N)

Классификация:

Стандартная кросс-энтропия:

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}},$$





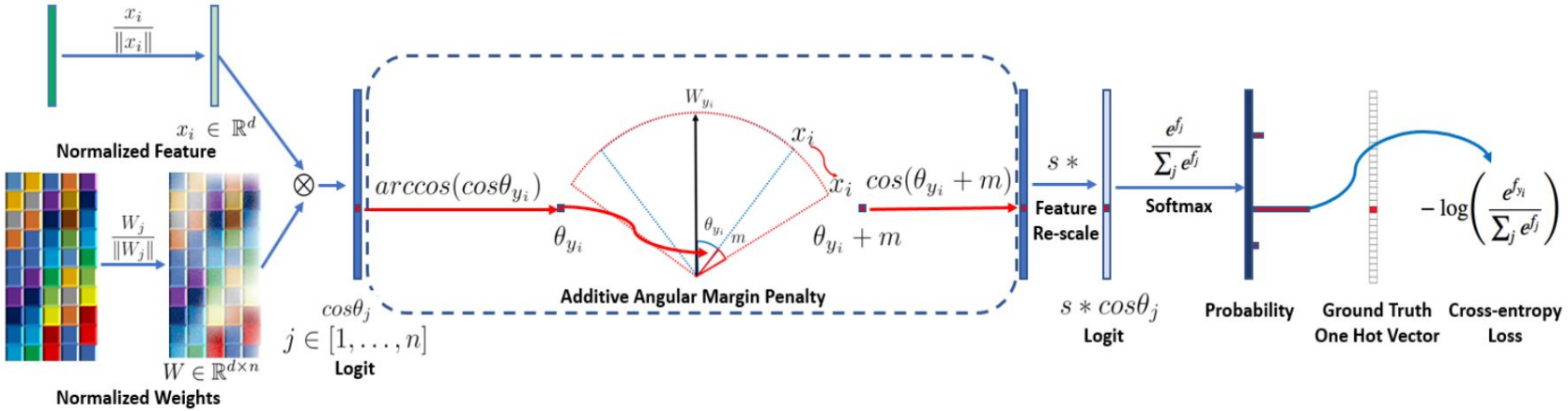
От Softmax к ArcFace

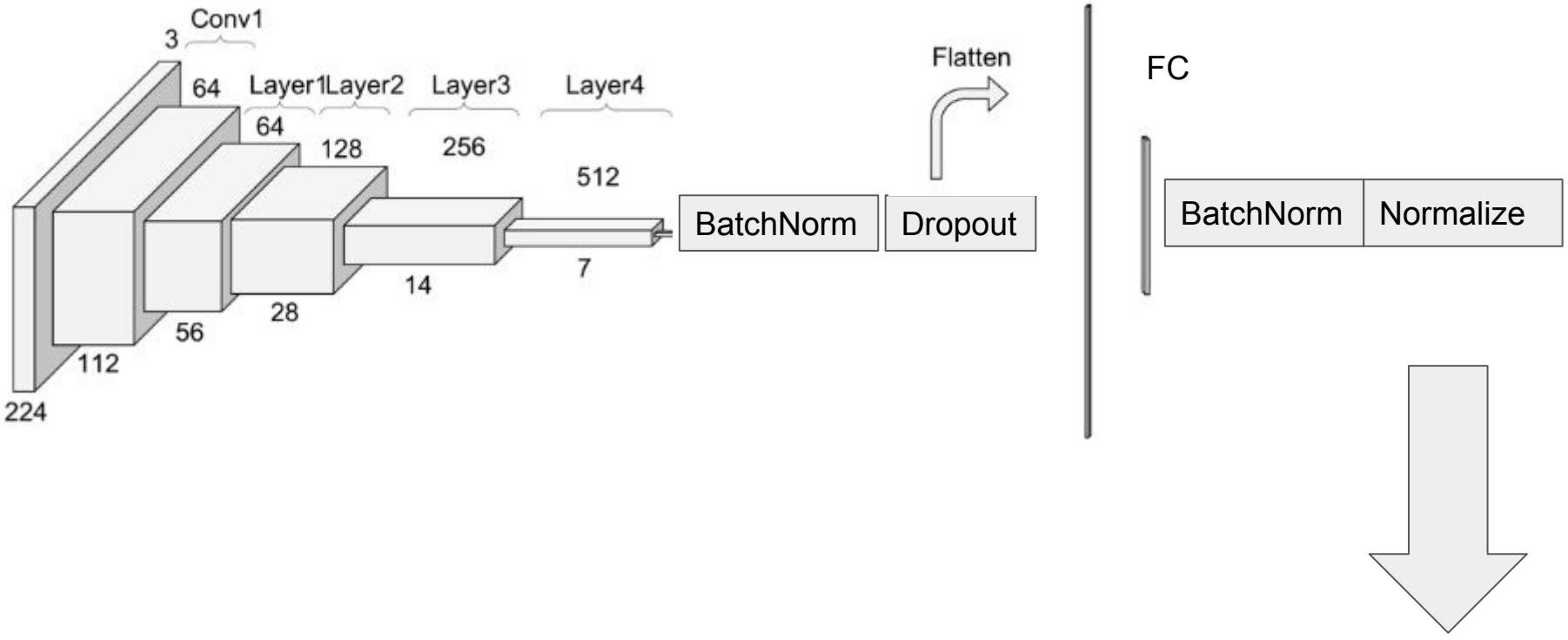
$$b_j = 0$$

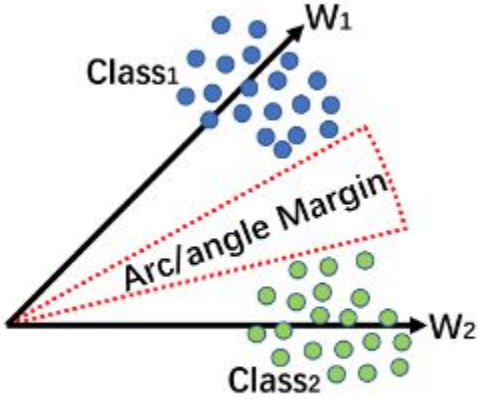
$$W_j^T x_i = \|W_j\| \, \|x_i\| \cos\theta_j$$

$$L_4 = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{e^{s(\cos(m_1 \theta_{y_i} + m_2) - m_3)}}{e^{s(\cos(m_1 \theta_{y_i} + m_2) - m_3)} + \sum_{j=1, j \neq y_i}^{n} e^{s \cos\theta_j}}$$

ArcFace


$\frac{x_i}{\|x_i\|}$
$x_i \in \mathbb{R}^d$
Normalized Feature
$\frac{W_j}{\|W_j\|}$
$W \in \mathbb{R}^{d \times n}$
Normalized Weights
$\otimes$
$cos\theta_j$
$j \in [1, \ldots, n]$
Logit
$arccos(cos\theta_{y_i})$
θ_{y_i}
W_{y_i}
x_i
θ_{y_i}
m
$cos(\theta_{y_i} + m)$
$\theta_{y_i} + m$
Additive Angular Margin Penalty
$s *$
Feature
Re-scale
$s * cos\theta_j$
Logit
$\frac{e^{f_j}}{\sum_j e^{f_j}}$
Softmax
Probability
Ground Truth
One Hot Vector
$-\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$
Cross-entropy
Loss


Conv1
3
64
Layer1
Layer2
Layer3
Layer4
64
128
256
512
224
112
56
28
14
7
Flatten
BatchNorm
Dropout
FC
BatchNorm
Normalize
ArcFace loss



(a) ArcFace

(b) Geodesic Correspondence

CNN

CNN

Descriptor 1

Descriptor 2

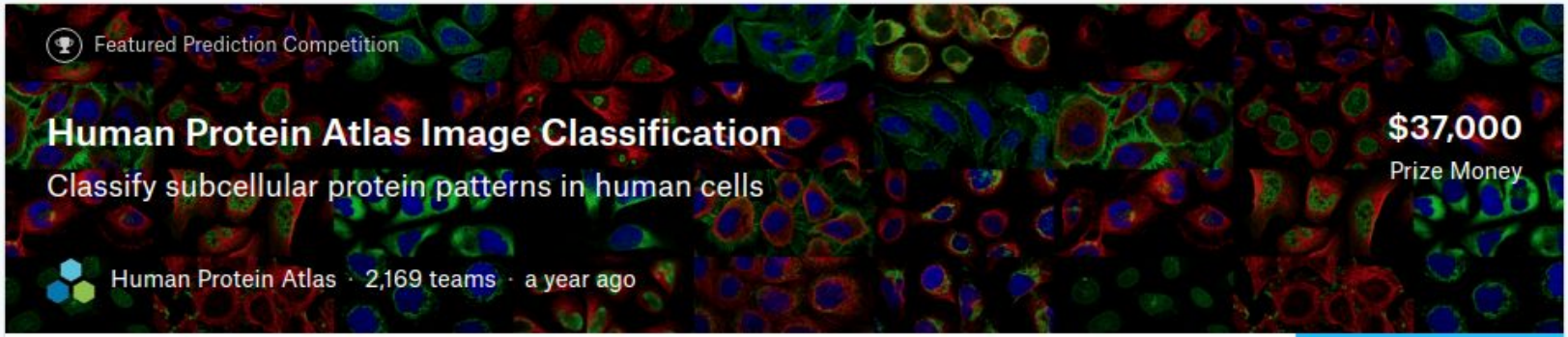
Similarity Score

Overview

After centuries of intense whaling, recovering whale populations still have a hard time adapting to warming oceans and struggle to compete every day with the industrial fishing industry for food.

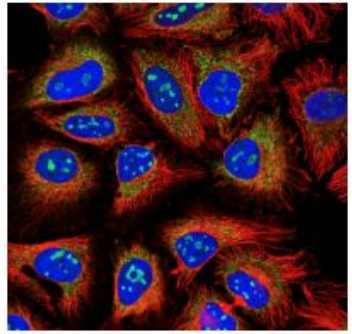
To aid whale conservation efforts, scientists use photo surveillance systems to monitor ocean activity. They use the shape of whales' tails and unique markings found in footage to identify what species of whale they're analyzing and meticulously log whale pod dynamics and movements. For the past 40 years, most of this work has been done manually by individual scientists, leaving a huge trove of data untapped and underutilized.

In this competition, you're challenged to build an algorithm to identify individual whales in images. You'll analyze



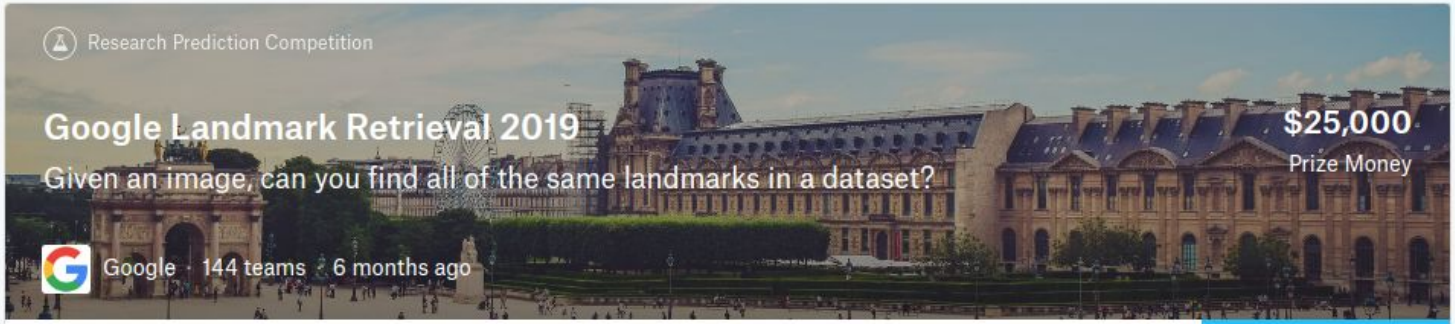
Overview

In this competition, Kagglers will develop models capable of classifying mixed patterns of proteins in microscope images. The Human Protein Atlas will use these models to build a tool integrated with their smart-microscopy system to identify a protein's location(s) from a high-throughput image.



Proteins are "the doers" in the human cell, executing many functions that together enable life. Historically, classification of proteins has been limited to single patterns in one or a few cell types, but in order to fully understand the complexity of the human cell, models must classify mixed patterns across a range of different human cells.

Images visualizing proteins in cells are commonly used for biomedical research, and these cells could



Overview

Image retrieval is a fundamental problem in computer vision: given a query image, can you find similar images in a large database? This is especially important for query images containing landmarks, which accounts for a large portion of what people like to photograph.

In this competition, Kagglers are given query images and, for each query, are expected to retrieve all database images containing the same landmarks (if any). The competition will proceed in two phases: The 1st phase will use the same test and index sets as last year, while for phase 2 we will release a completely new dataset that contains 700K images with more than 100K unique landmarks. We hope that this release will accelerate progress in this important research problem.

This challenge is organized in conjunction with the Landmark Recognition Challenge (https://www.kaggle.com/c/landmark-recognition-2019). In particular, note that the test set for both challenges is the same, to encourage participants to compete in both. We also encourage participants to use the training data from the recognition challenge (either from this year's or last year's dataset) to develop models which could be useful for the retrieval challenge.

```python
class ArcMarginProduct(nn.Module):

    def __init__(self, in_features, out_features, s=30.0, m=0.50, easy_margin=False):
        super(ArcMarginProduct, self).__init__()
        self.in_features = in_features
        self.out_features = out_features
        self.s = s
        self.m = m
        self.weight = Parameter(torch.FloatTensor(out_features, in_features))
        nn.init.xavier_uniform_(self.weight)

        self.easy_margin = easy_margin
        self.cos_m = math.cos(m)
        self.sin_m = math.sin(m)
        self.th = math.cos(math.pi - m)
        self.mm = math.sin(math.pi - m) * m

    def forward(self, input, label):
        # --------------------------- cos(theta) & phi(theta) ---------------------------
        cosine = F.linear(F.normalize(input), F.normalize(self.weight))
        sine = torch.sqrt((1.0 - torch.pow(cosine, 2)).clamp(0, 1))
        phi = cosine * self.cos_m - sine * self.sin_m
        if self.easy_margin:
            phi = torch.where(cosine > 0, phi, cosine)
        else:
            phi = torch.where(cosine > self.th, phi, cosine - self.mm)
        # --------------------------- convert label to one-hot ---------------------------
        # one_hot = torch.zeros(cosine.size(), requires_grad=True, device='cuda')
        one_hot = torch.zeros(cosine.size(), device='cuda')
        one_hot.scatter_(1, label.view(-1, 1).long(), 1)
        # -------------torch.where(out_i = {x_i if condition_i else y_i) -------------
        output = (one_hot * phi) + ((1.0 - one_hot) * cosine)  # you can use torch.wher
        output *= self.s
        # print(output)

        return output
```

https://github.com/ZhaoJ9014/face.evoLVe.PyTorch