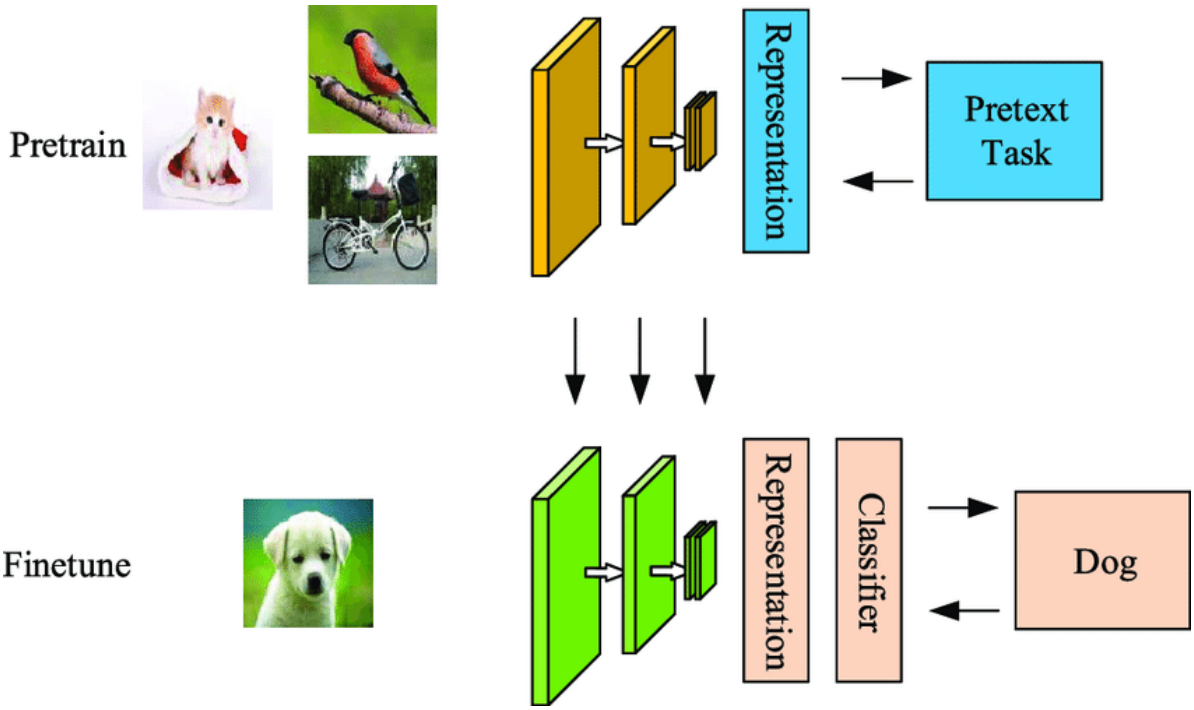
介绍:

公元2020年，一个瘟疫肆虐的时代，当大家都在惶恐时，在AI领域 self-supervised learning正在悄然发展.... self-supervised learning 近年来成为炙手可热的话题。 大佬yann在许多公开场合宣传self-supervised learning 才是AI的未来。 hinton，kaiming 等大佬在顶会上发表的simclr 和moco系列性能直逼supervised learning。 似乎self-supervised learning 是引领 AI领域的下一个激浪。这让学者们对这个新鲜的学习范式充满好奇，究竟什么是self-supervised learning？ 为何它能有如此惊人的效果。在这篇文章中，我会解释self-supervised learning，并对当前的主流方法分类介绍， 对其背后的原理给出一些浅显的解释。其中不乏纰漏，希望各位批评指正，不胜感激!

Self-supervised learning & supervised-learning

我们最常接触到的学习范式是supervised-learning（监督学习）。 这种方式需要标注数据，即训练数据中需要有input data 和label。 在监督学习范式下模型会学到一种映射，能够将输入数据映射成label。 比如最常见的image classfication. 输入一张图片，让模型输出这张图片的类别。其中我们认为标签就是监督信息，来监督模型学习数据到标签的映射。 而self-supervised learning 并不需要标注数据，但同样有监督信息。 它的监督信息直接通过数据产生。

有人认为self-supervised learning属于 un-supervised learning，因为两者都使用unlabel data，而有人认为 self-supervised learning属于supervised learning，因为两者都提供监督信息，un-supervised learning 通过聚类学习，无需监督信息。 目前学术界没有主流的说明，而是把self-supervised learning 算为单独的一种学习范式。



该图表示用自监督学习的方式在pretxt task上预训练，然后在下游任务（downstream task）分类上微调（finetune）。

self-supervised learning的任务和supervised learning的任务不同。 前者希望模型能够学习数据中隐含的semantic information，让模型学习到一种表示（representation），该表示中包含原始数据中的所有语义特征，后者只需要模型能够将数据映射成标签，正确分类图片即可，并不在意数据中的特征。 但模型中的semantic information对分类等supervised learning task是有帮助的。 如果模型能学习到图片中底层的特征，就能更好的区别不同的图片从而提升分类的准确度。 比如分类汽车和自行车，如果模型能够发现汽车都是四个轮子的特征，而自行车是两个轮子的特征，就能更好的区别这两类。 并且在supervised learning task中，学者发现神经网络的浅层部分一般用来捕获semantic information，

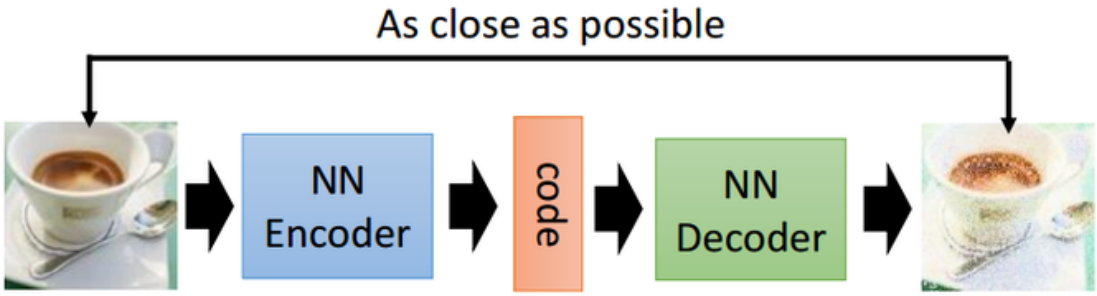
深层部分则会利用这些底层的semantic features 来抽取与task相关的特征。 这与self-supervised learning有异曲同工之妙。 但self-supervised learning的优点不止于此。 真实世界中存在一些难以标注的数据。比如医疗图片，正样本的数量很少，自然标注信息少，或是NLP数据， 文本的数量一般很大，最大的有几T数据，人工标注显然不现实。缺少标注信息就很难建立一个supervised learning model。而这时通过self-supervised learning来学习数据的semantic information，再通过少量样本进行supervised learning 效果肯定会比直接进行supervised learning好。 而这样的模式用专业术语描述为： 先用self-supervised learning pre-train 模型，再在downstream task上fine-tune。**总结一下，self-supervised learning的任务是让模型学习到数据的semantic information 从而更好的完成supervised learning task。**

self-supervised learning 的分类

self-supervised learning 的方法可以分为两大类：1. generative methods 2. discriminative methods。 generative methods 通过对数据进行高细粒度的重建来学习数据中的semantic information。 discriminative methods 中典型的 contrastive method 通过构建正负样本，依据正例样本的相似性和负例样本差异性让模型学习数据中的底层特征。 最近 Hinton发表的 SIMCLR 和 Kaiming 的 MOCO都是基于contrastive method完成的。 效果十分亮眼。 因此这篇文章的主要重心也是Contrastive learning。

Generative model

generative model的基本思路是：先将数据编码成一个向量，再对其进行解码重建成原数据。 通过这个过程学习数据中的语义信息。 以图片为例，先将图片编码成一个向量，再解码重建回原图。 如果重建的图和原图足够接近，就认为模型学会了如何抽取图片中的语义信息。 如图是VAE（variational auto-encoder）的基本架构，其中Encoder将输入数据编码成向量cde，decoder将向量解码，重构成原数据。



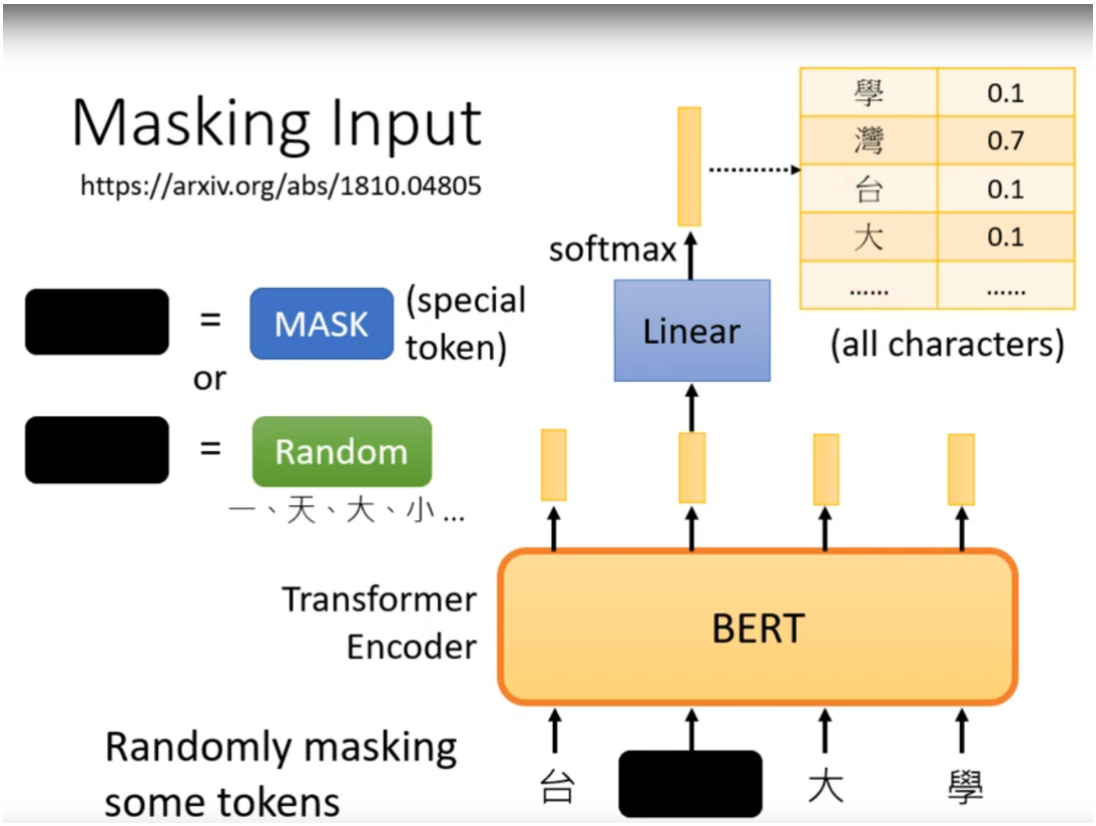
可以看到，重建的图片和原图相比是有点糊的，这是因为重建图片是一个pixel-wise的任务，而encoder将输入的高维图片压缩成一个低维的向量，其中一定会有信息的损失，完全重建出每个pixel的信息是十分困难的。 其次，如果图片的足够大，这样的重建开销也是很大的。

最关键的是，我们并不关心每个pixel的信息，我们关心的是这张图片中的主要特征，这些特征就能完美地表达原图中的信息，从而帮助我们完成下游任务。 这个例子中，我们通过记忆里的信息只画出一张dollar的大致轮廓而不是所有细节，但这并不影响我们区别真币和假币。 这说明，一些主要的特征就足够表达数据的特征，一些细节并没有什么用。

Fig. LEFT: Drawing of a dollar bill from memory. RIGHT: Drawing subsequently made with a dollar bill present. Image source: [Epstein, 2016]

discriminative methods

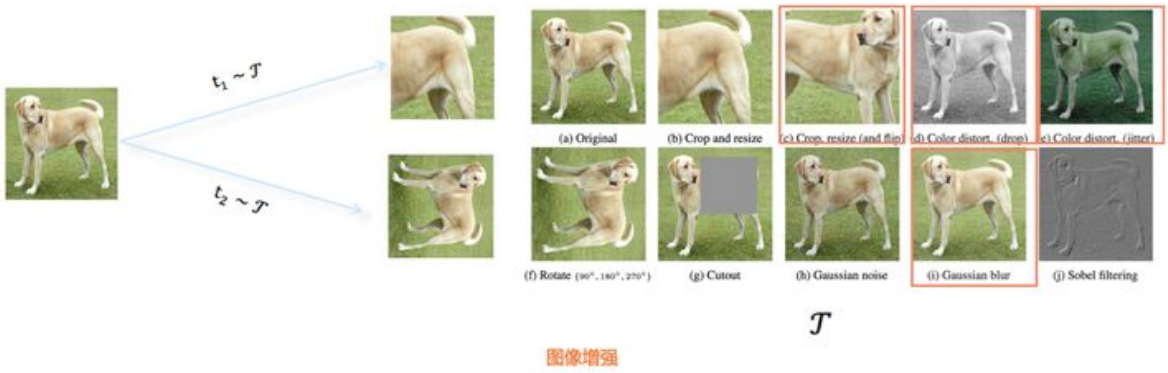
discriminative methods 通过区别不同实例让模型学习其背后的语义信息。 discriminative methods不仅包括现在火热的contrastive learning，在NLP领域，也有很多应用。 比如对输入的句子随机mask一个词语，让模型预测出被mask的词。



contrastive learning

为什么 contrastive learning work？ 我们知道self-supervised learning需要从数据中提供监督信息，

contrastive learning是discriminative method的代表，现在几乎成为self-supervised learning的代名词了。 contrastive learning通过data augmentation 构造相似实例和不相似实例（即同一数据经过不同augmentation 生成的实例是相似的，而不同实例间互不相似），要求模型将实例编码成一个表示（representation），使得相似实例在该表示空间距离近，而不相似实例间距离远。 这就是contrastive learning 中的 代理任务（proxy task），通过完成这个任务使得模型学会将数据编码成有效的representation。 其中相似实例称为正样本，不相似实例称为负样本。 通过拉进相似实例之间的距离，模型能够学到某一样本独特的特征，通过拉远不相似实例间的距离，模型能学到不同样本之间的区别。 在contrastive learning 存在一个模型坍塌（model collapse）的问题。模型坍塌是很致命的问题，会导致我们学到的表示无效。 如果我们的data augmentation 构造的实例和原图相差过大，比如把狗图片中的狗crop掉，只剩下草地。 这时该实例在语义上已经不在是狗，还认为该生成的实例和原图狗是相似样本就会导致模型学习到错误的语义信息。 而如果data augmentation的效果过弱，就会导致生成实例与原数据很相似，模型无法学习到有效的表示。 过强或过弱的augmentation 都会导致模型坍缩。除此之外还有其他因素也会导致坍缩，我会在下文谈到。 因此如何生成高质量的相似样本，以及如何防止模型坍缩是contrastive learning 领域中主要的研究问题，以下提及的论文都会从这两个角度切入。



SimCLR A Simple Framework for Contrastive Learning of Visual Representations

SimCLR是Geoffrey Hinton在2020年发表的一篇文章。如图所示，其效果十分惊人，基本上把其他baseline踩在脚底。 随着模型参数的提高， SimCLR的效果也在不断提升，当参数达到400M时，效果已经接近Supervised learning。

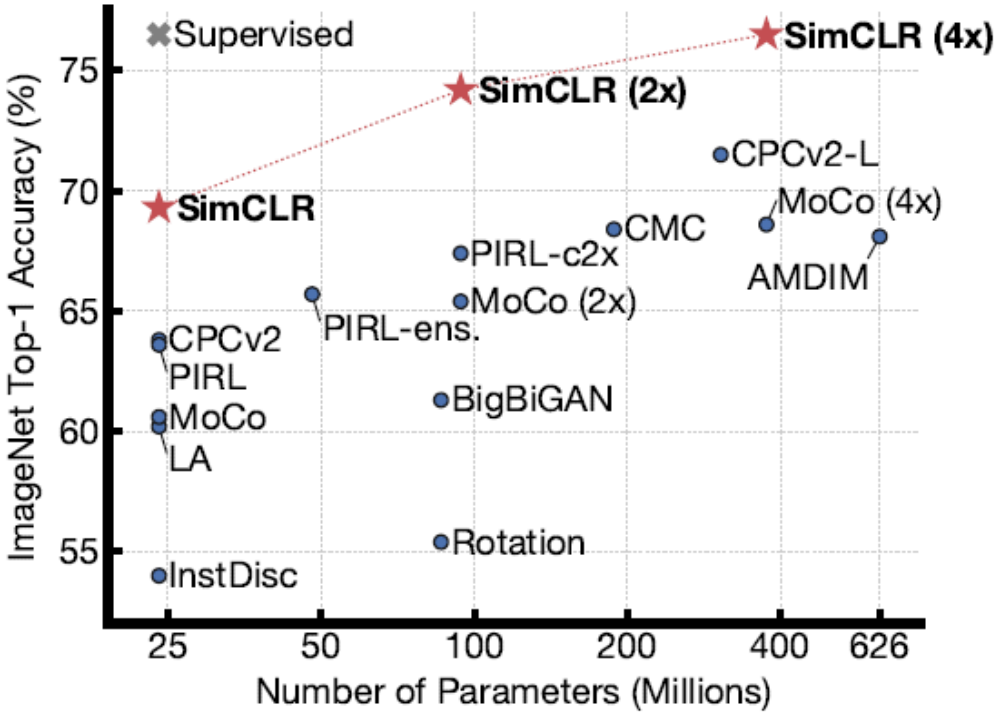


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

在SimCLR的论文中，作者提出4个结论：

1. 组合的数据增广能够产生高质量的相似样本使得模型能在proxy task充分训练，从而让产生有效的representation。
2. 在representation和loss之间增加一个非线性变换能提高representation的质量。
3. normalized embedding和合理的温度参数（temperature parameter）对contrastive learning的损失函数有益。
4. 大batch size， 大参数量对 contrastive learning有益。

一个简短的Video 介绍了SimCLR的工作原理：Link

下图是论文中给出的framework：

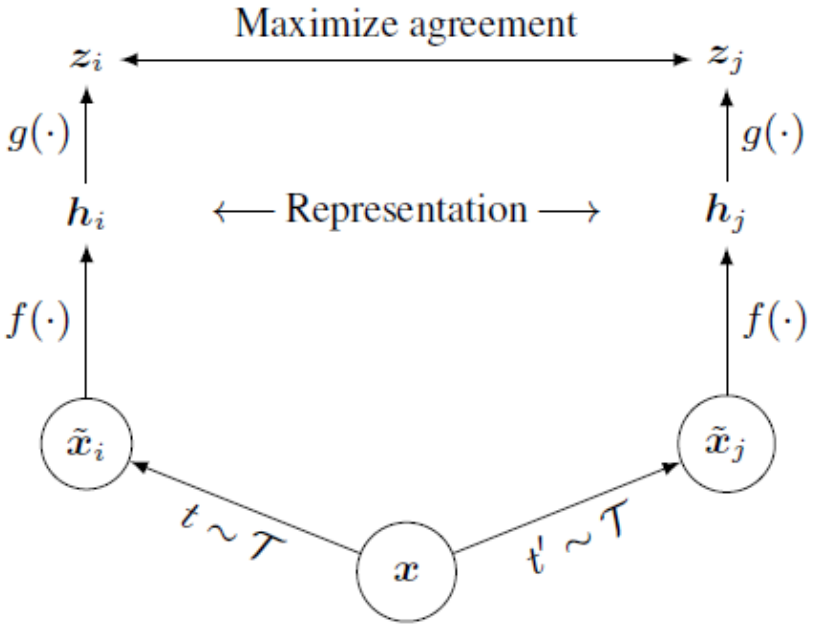


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and use encoder $f(\cdot)$ and representation $\boldsymbol{h}$ for downstream tasks.

基本步骤如下：

1. 通过data augmentation 生成同一图片的相似样本xi，xj， 同一图片不同augmentation得到的不同样本。
2. 用encoder **f()** 对xi，xj编码成representation h.
3. 用decoder **g()** 对 h 解码成z。
4. 通过损失函数，使得zi和zj 间距离接近。

其中data augmentation包括：random cropping followed by resize back to the original size, random color distortions, and random Gaussian blur 以及他们的随机组合。

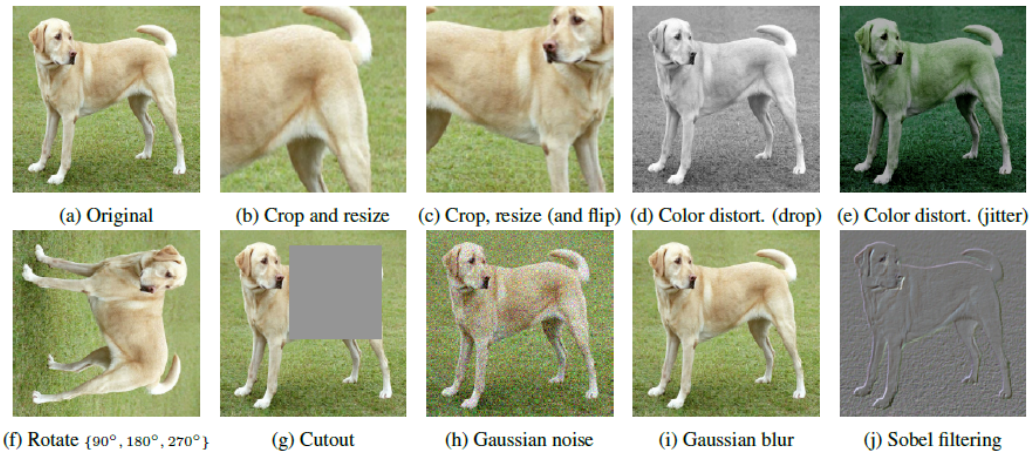


Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we *only* test these operators in ablation, the *augmentation policy used to train our models* only includes *random crop (with flip and resize)*, *color distortion*, and *Gaussian blur*. (Original image cc-by: Von.grzanka)

根据作者的研究不同组合的augmentation 效果不同。 其中对角线是单一augmentation，效果都很差，而color和crop组合的效果最好。

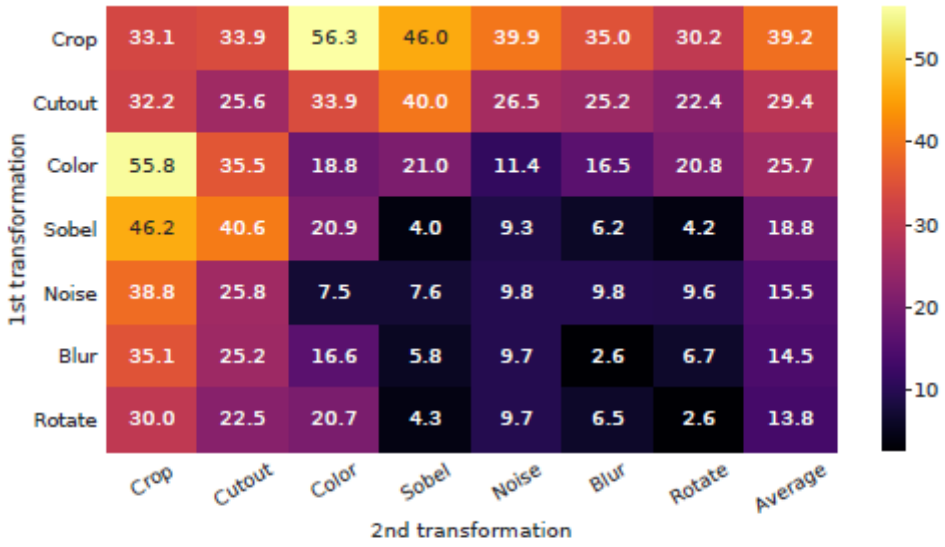


Figure 5. Linear evaluation (ImageNet top-1 accuracy) under individual or composition of data augmentations, applied only to one branch. For all columns but the last, diagonal entries correspond to single transformation, and off-diagonals correspond to composition of two transformations (applied sequentially). The last column reflects the average over the row.

框架中的encoder是resnet.

decoder部分（论文中称projection）使用MLP实现。具体地： **$z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$**

损失函数： 给定{Xk} 是生成样本集， 由于每个输入样本会通过两个 augmentation 生成两个正样本，则 对于 min-batch length = N 的数据集，{Xk} 的长度为2N。 对于一个输入样本，它对应有两个正样本记为zi zj，剩下2N-2个样本为负样本。

对于i j 两个正样本，他们定义的损失函数如下：

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\mathrm{sim}(z_i, z_k)/\tau)}, \qquad (1)$$

其中 sim(zi,zj)是余弦相似度（cosine similarity），τ是温度系数，现在先忽略，后面会详细解释它。

通过最小化损失函数，即增大分子，就能增大两个正样本i j 之间的相似性，减少分母，减少正样本i与其余负样本的之间的相似性。

完整的损失函数如下图所示：

$$\textbf{define } \ell(i,j) \textbf{ as } \ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(s_{i,k}/\tau)}$$
$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$$

对于作者提出的结论2：在representation和loss之间增加一个非线性变换能提高representation的质量。 作者做了如下实验，在不同dimensionality情况下，使用三种projection，None （不使用projection）， Linear， and Non-linear。发现不论dimensionality如何变换，Non-linear效果最好，而不使用Projection效果下降很大。

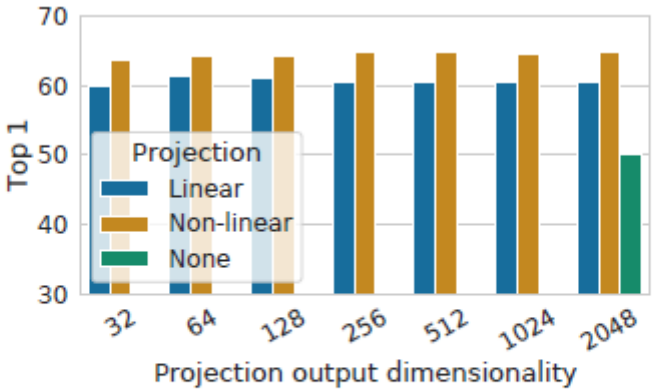


Figure 8. Linear evaluation of representations with different projection heads $g(\cdot)$ and various dimensions of $z = g(h)$. The representation h (before projection) is 2048-dimensional here.

我认为这主要是底层通过encoder学习到的都是原始图片的底层语义信息，包含原图中的所有特征。 其中必然有些特征对对比不同样本没有帮助。 这些task 无关的特征对任务本身来说是噪声，会影响模型的判断。 因此通过一个非线性层将task无关的信息过滤掉，只保留能够区别不同样本的特征，更好的完成proxy task本身。 而线性层效果比非线性层要差，这是因为线性层只能做线性变换，经过线性映射的h 还是被投影回原空间，无关信息无法被过滤。

为了证明是否真的 representation h中有更加丰富的语义信息，而最终用于完成任务的z = g(h) 更少， 作者做了如下实验： 用h 和 g（h） 分别去预测使用的是哪种 augmentation method，结果表示：使用h预测效果更好。 说明 h中保留了更多的底层图片信息。

What to predict?	Random guess	Representation h	Representation $g(h)$
Color vs grayscale	80	99.3	97.4
Rotation	25	67.6	25.6
Orig. vs corrupted	50	99.5	59.6
Orig. vs Sobel filtered	50	96.6	56.3

Table 3. Accuracy of training additional MLPs on different representations to predict the transformation applied. Other than crop and color augmentation, we additionally and independently add rotation (one of $\{0°, 90°, 180°, 270°\}$), Gaussian noise, and Sobel filtering transformation during the pretraining for the last three rows. Both h and $g(h)$ are of the same dimensionality, i.e. 2048.

结论4： 大batch size， 大参数量对 contrastive learning有益。

batch size 越大，同一个batch中负样本的数量越多，自然学习到的样本越多。效果越好。 而越大的参数量意味着模型有更大的学习空间， 同时self-supervised learning 能通过 data augmentation提供足够多的数据学习，效果自然会更好。

如图，随着参数量的提高，self-supervised model（红线）的效果不断提高，而监督学习的模型（绿线） 在后期增长变缓。 主要是因为监督学习的label data有限。

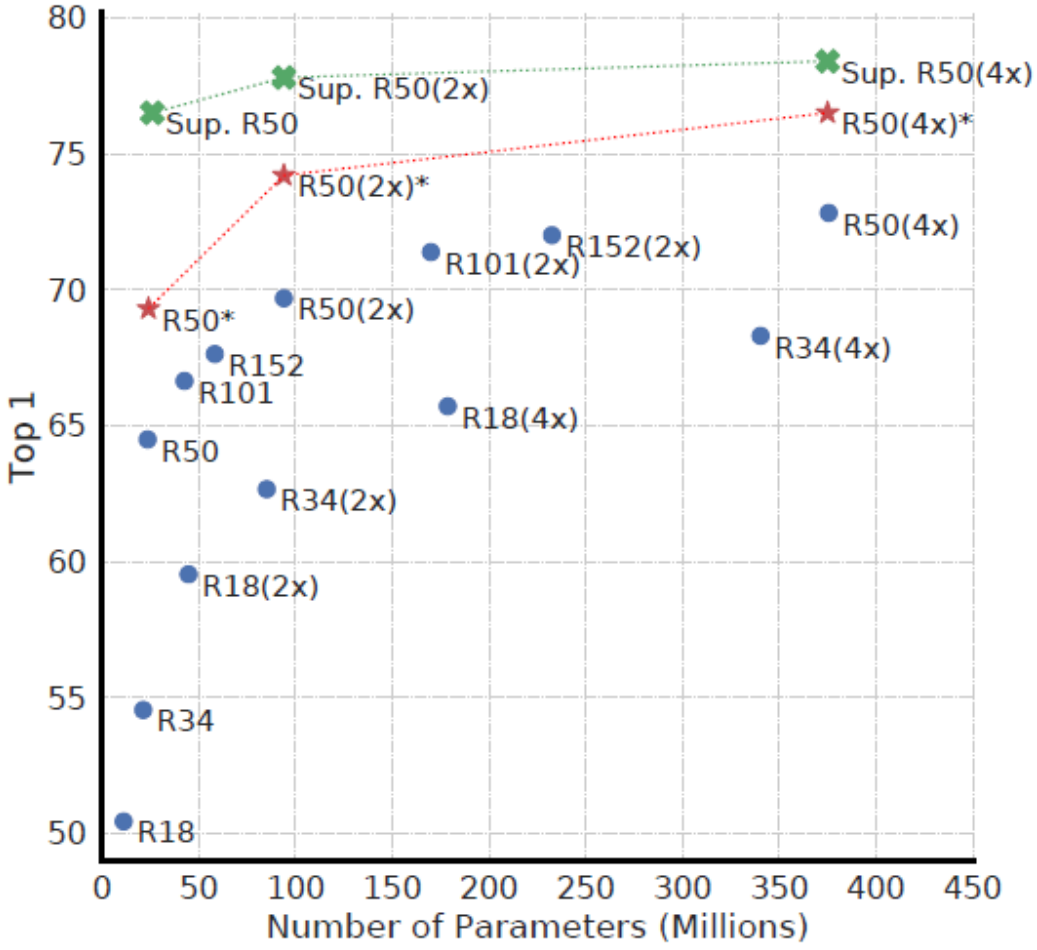
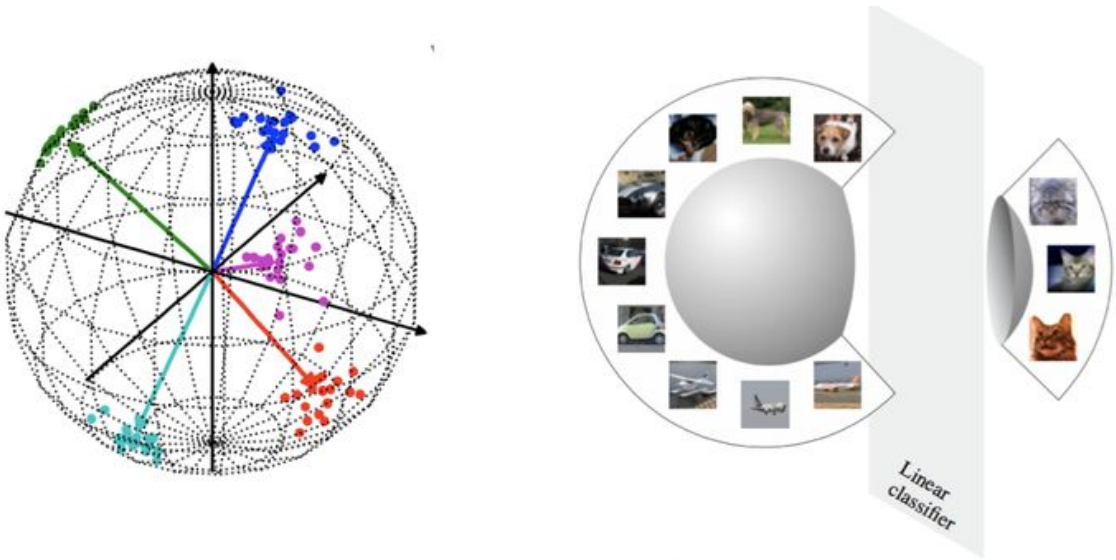


Figure 7. Linear evaluation of models with varied depth and width. Models in blue dots are ours trained for 100 epochs, models in red stars are ours trained for 1000 epochs, and models in green crosses are supervised ResNets trained for 90 epochs[7] (He et al., 2016).

contrastive learning 究竟干了啥?

前面我们计算相似性时使用的是余弦相似性，其实也可以理解为做了L2 Norm 的向量内积。两者公式是一致的。

这样就可以认为，得到的z 是处于超球面的一个单位向量，其长度为1.

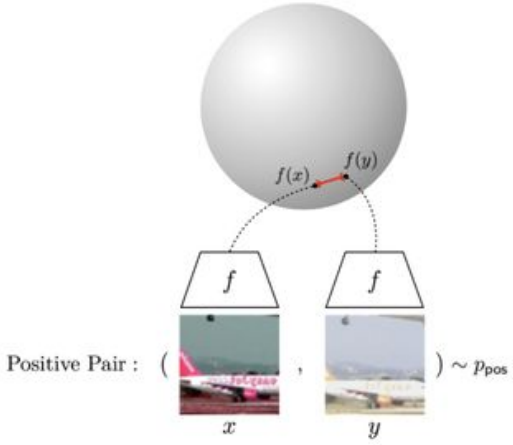


单位超球面及线性可分

这样的好处是：

1. 经过normlization后损失函数更加光滑，对优化有好处。
2. 相同样本的向量 z 会聚集于超平面的同一个区域。

根据这篇论文“Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere”， 一个好的contrastive learning需要做到两点： 1. Alignment 2. Uniformity。

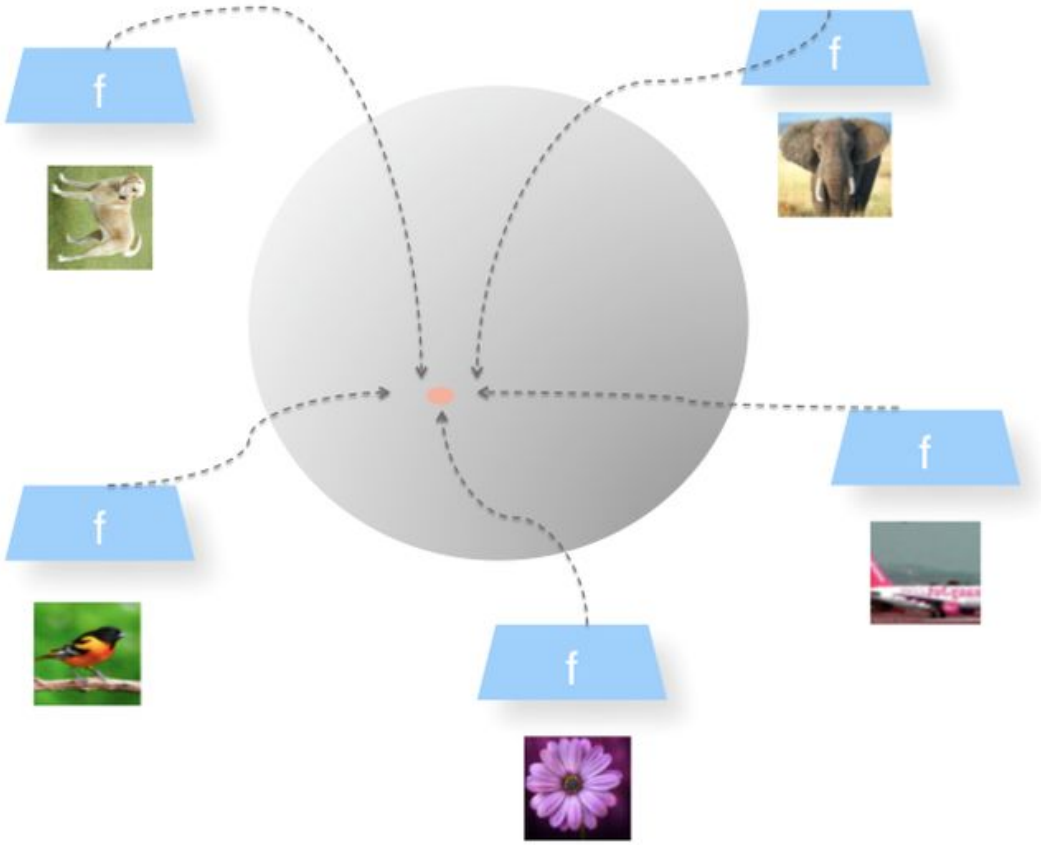


Alignment：相似实例有相近的特征　　Uniformity：保留尽可能多的信息

From:Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere

Alignment即是相似样本间距离要足够近，而Uniformity 通过拉远负样本之间的距离，使得不同类样本均匀分布在该超球面上。

如果一点也不Uniformity 即 所有样本都处于超球面上的同一点，那这样的representation无法给我们带来任何discriminative information。



模型的坍塌

两个不同输入样本其实属于同一个语义信息，却被当做负样本怎么办?

survey effective augmentation methods

温度系数τ是什么？ 有什么用?

根据论文“Understanding the Behaviour of Contrastive Loss”, 发现τ是一个很重要的超参数，对模型的表现影响很大。

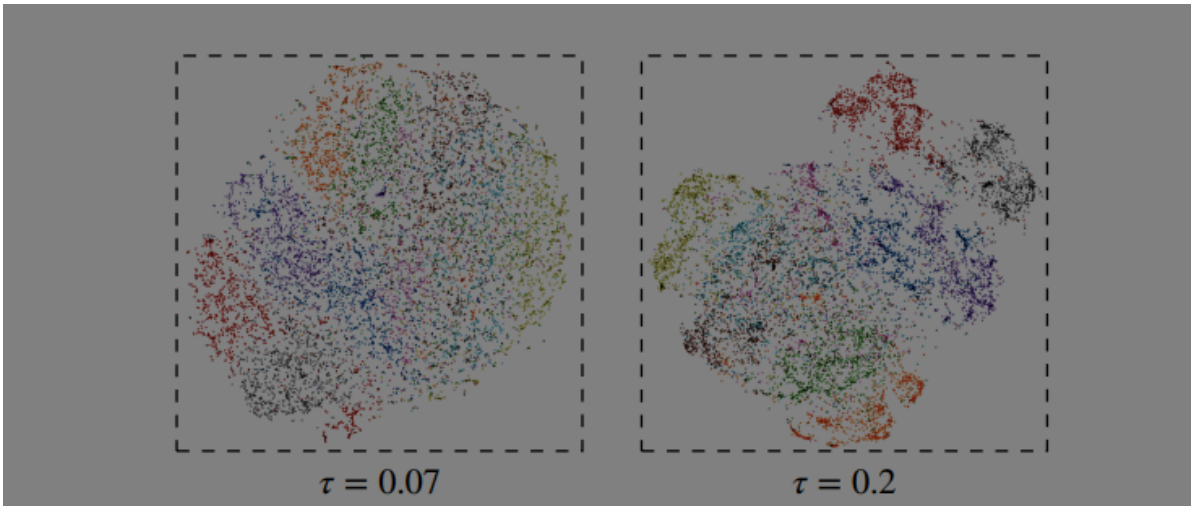


Figure 2. T-SNE [29] visualization of the embedding distribution. The two models are trained on CIFAR10. The temperature is set to 0.07 and 0.2 respectively. Small temperature tends to generate more uniform distribution and be less tolerant to similar samples.

一个合适的τ(0.07 left) 能够均匀地划分样本。说明这时模型能够区别负样本之间的差异，识别正样本间的相似性，真正学习到了数据的语义信息。

那τ究竟在损失函数中发挥着什么作用?

直观地看， sim(xi, xj) = s ∈ [-1, 1] , then exp(sim(..)) = exp(s) ∈ [1/e, e] , 而分母是 sum(exp(s)) over batch， 而一般batch size 取得很大 ≈1000 这个数量级，这就会导致 损失函数很小，通过温度系数把 s 缩放到一个合适的范围，使得对损失函数的优化更加稳定。

其次，τ能将模型优化的重点聚焦于有难度的负样本（hard sample)上，加大对它们的惩罚。 何谓hard sample? 即在超球面中与正样本相近的负样本。 何谓惩罚？即对这些hard sample 分配更高的权重，拉远它们与正样本的距离。 可以想象成 正负样本间有斥力，距离近的样本会被τ赋予更大的斥力来相互排斥。 如果τ很小，则在正样本附近的负样本就会被认为是有难度的负样本，被赋予更大的斥力。而τ很大，则斥力会被分配到更大范围内的负样本中，关注到的负样本数量增多，此时每个负样本受到的平均斥力就会更小。 按照这个理论，τ能够将负样本推远，将密集分布的样本打开，使得全部样本在超球面上均匀分布。 如上图所示。 当然τ不是越小越好， 如果τ接近于0，则只关注于正样本附近最近的一两个负样本。而在训练中，输入的正样本和负样本可能在语义上是同一类，虽然他们被模型认作负样本，但在超球面中，它们的距离应该是相近的。 此时如果模型强行将两者拉远，会使得模型学到错误的语义信息。 在SimCLR的框架中， 由于我们无法提前知道每张图片的标签，因此很可能出现这种情况：同属于狗的两张图片，被当做负样本。 因此这种情况在训练中会经常发生，通过找到合适的τ 就能在一定程度上缓解这个问题。 但想完全解决，还需要参考这篇论文： 。

Self-supervised learning 的应用

在前文我们说过 self-supervised learning 的任务是通过设置合理proxy task 让模型学到数据的representation（对于contrastive learning来说是设计好的augmentation），从而在下游任务中达到更好的表现。 proxy task的好坏直接决定学得的representation 的质量。 对于不同领域来说，self-supervised learning这种学习的范式是不会变的。 针对不同领域中的数据，需要具体设计对应的proxy task。 因此，以下介绍的应用场景都将围绕 proxy task的设计。

更多设计proxy 的技巧：(27 封私信) 你见过哪些新颖的或有效的「自监督学习样本构建技巧」？ - 知乎 (zhihu.com)

Self-supervised learning In NLP

这篇文章几乎都是以 图片为例来解释self-supervised learning，而事实上self-supervised learning 应用最广的领域是 NLP。 NLP领域中的数据：文本，数据量极大（多达几T），这么大的数据几乎无法提供标签。 如此丰富的数据非常适合从数据中提取监督信息对模型进行训练，从而学到有价值的 representation 供下游任务中使用。

在NLP中进行对比学习，关键是设计好的augmentation methods，比如 Cutoff (reference)

Cutoff

Shen et al. (2020) proposed to apply **Cutoff** to text augmentation, inspired by cross-view training. They proposed three cutoff augmentation strategies:

1. *Token cutoff* removes the information of a few selected tokens. To make sure there is no data leakage, corresponding tokens in the input, positional and other relevant embedding matrices should all be zeroed out.,
2. *Feature cutoff* removes a few feature columns.
3. *Span cutoff* removes a continuous chunk of texts.

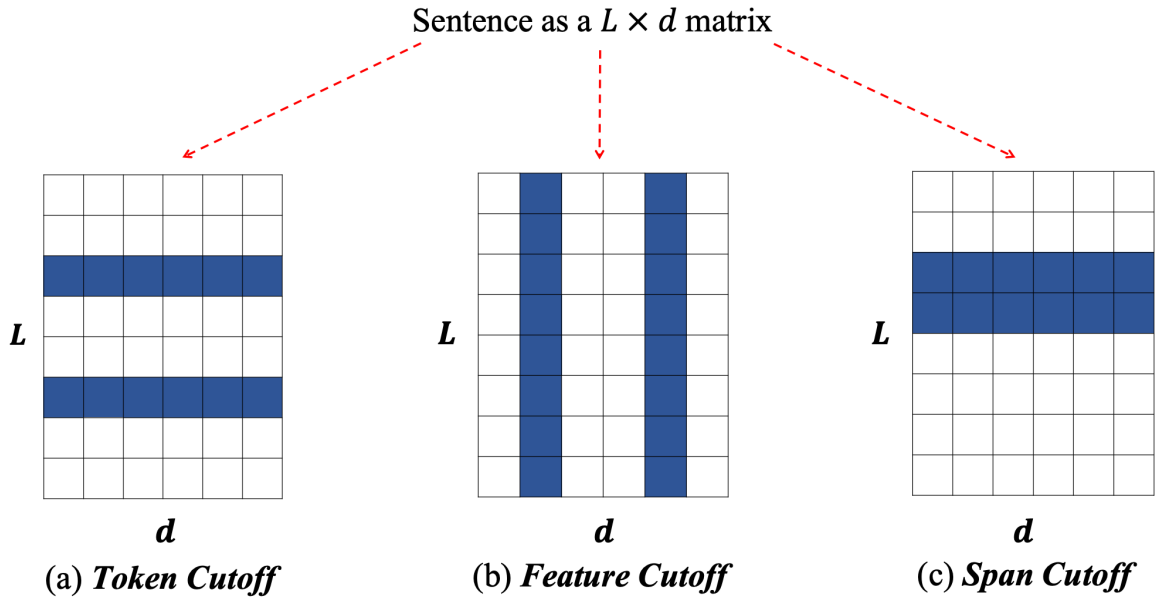
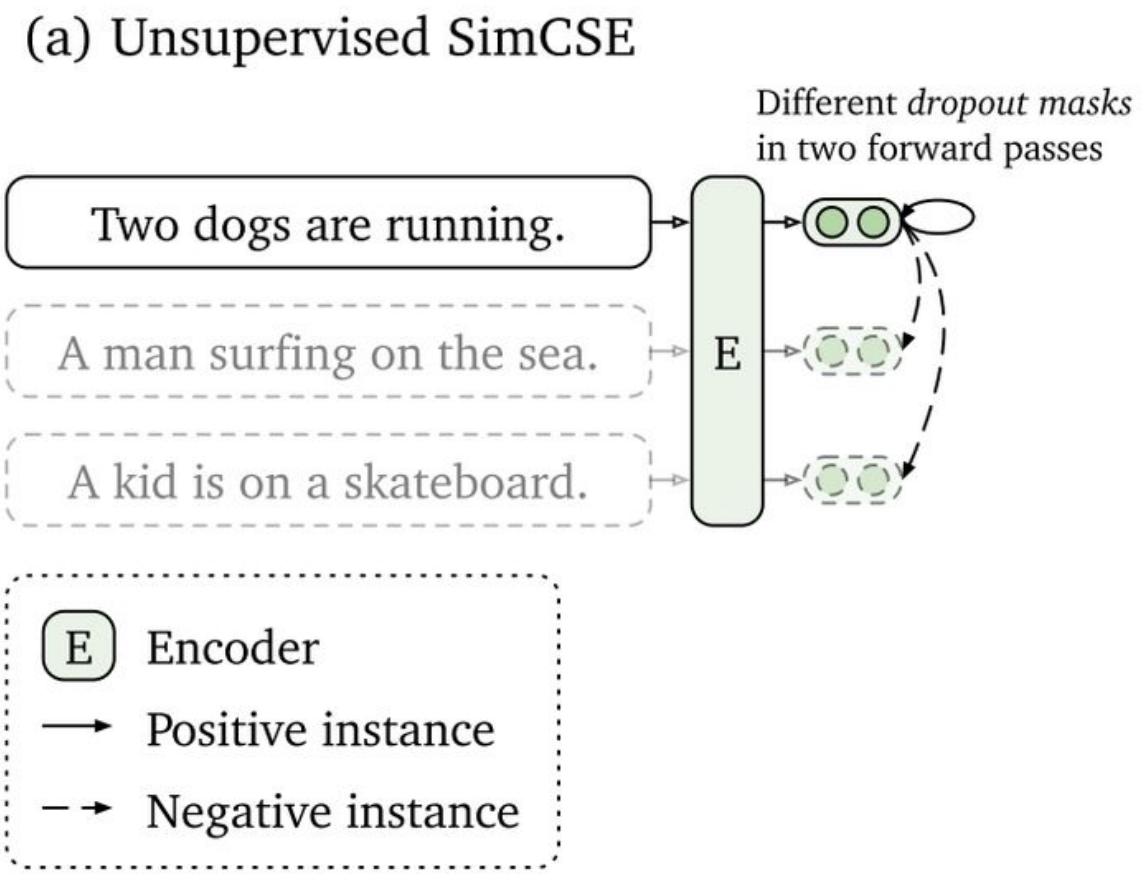


Fig. 21. Schematic illustration of token, feature and span cutoff augmentation strategies. (Image source: Shen et al. 2020)

详细内容可以参考我的这篇笔记：《自监督学习：IN NLP 从芝麻街开始的故事》。 里面除了介绍 contrastive learning，也介绍了在BERT中使用的其他自监督学习方法。

在这里我介绍一种Contrastive leaning的方法：SimCSE



SimCSE通过模型中dropout层随机关闭神经元，因此每次模型forward pass时都会生成两个不同的子模型，那么同一样本两次输入得到的token就会不同，而两次输入得到的token就是生成的正样本对，同一个batch中的其他样本则会被看做负样本对。

Self-supervised learning In Video

与图片不同，Video是一种更加复杂的数据，它可以看做一系列连续的图片。因此video不仅考虑空间特征，还需要考虑时间特征以及时空特征（spatial-temporal features). 正因为video有更多维度上的特征，我们才能从不同角度挖掘video数据中的监督信息。更加详细的解释可以参考：[基于视频的自监督学习 - 知乎 (zhihu.com)]

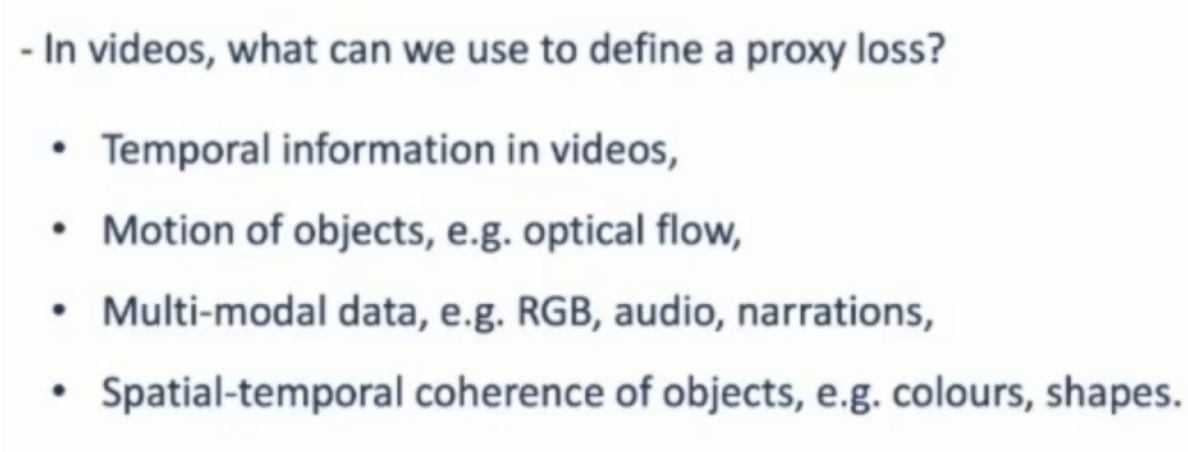


Image source：[计算机视觉 - 自监督学习 - Self-supervised Video Representation Learning*哔哩哔哩*bilibili]

Motion-based Proxy task

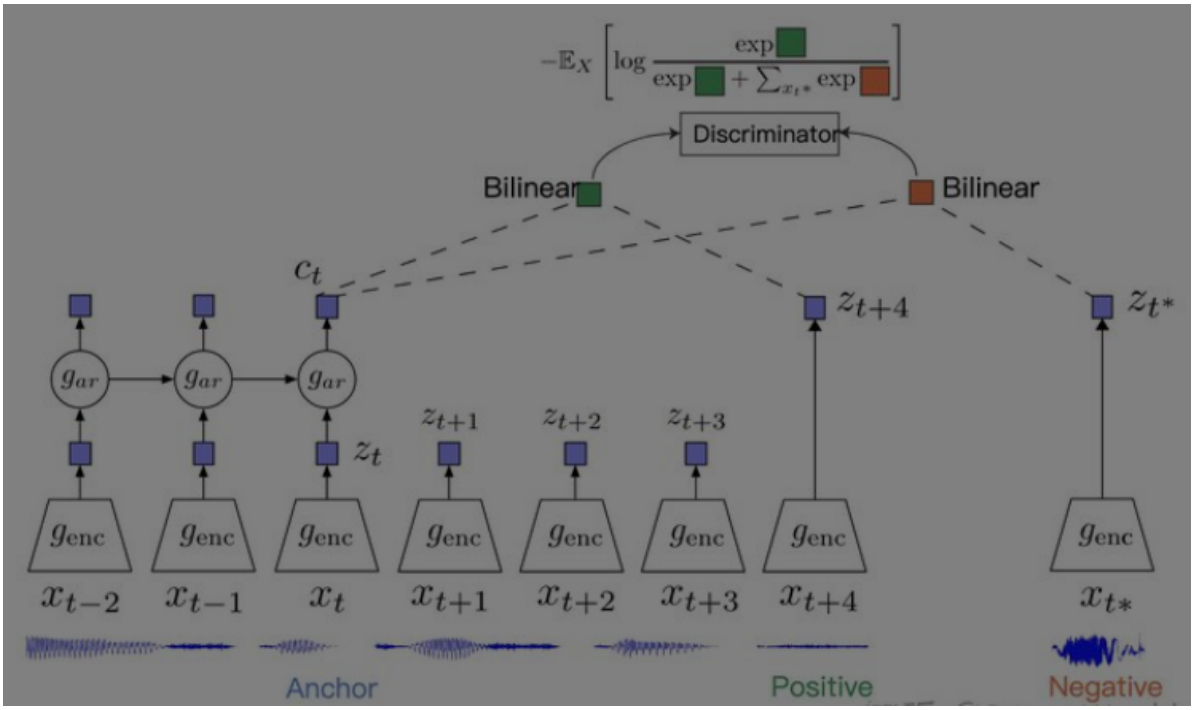
视频与图片最大的区别是，视频是连续运动的，因此视频中的运动是非常明显的特征。 通过光流聚类相似动作的样本，构造正样本对 用于contrastive learning。

Context-based Proxy task

视频在时间上是连续的，前后两帧的信息在语义上相关。 因此我们能够基于上下文来构造正样本对进行contrastive learning。

Representation Learning with Contrastive Predictive Coding: **CPC**

对于序列数据， 将给定窗口范围内的数据当做正样本对 （postive pair），从输入序列中随机采样一个样本作为负样本。



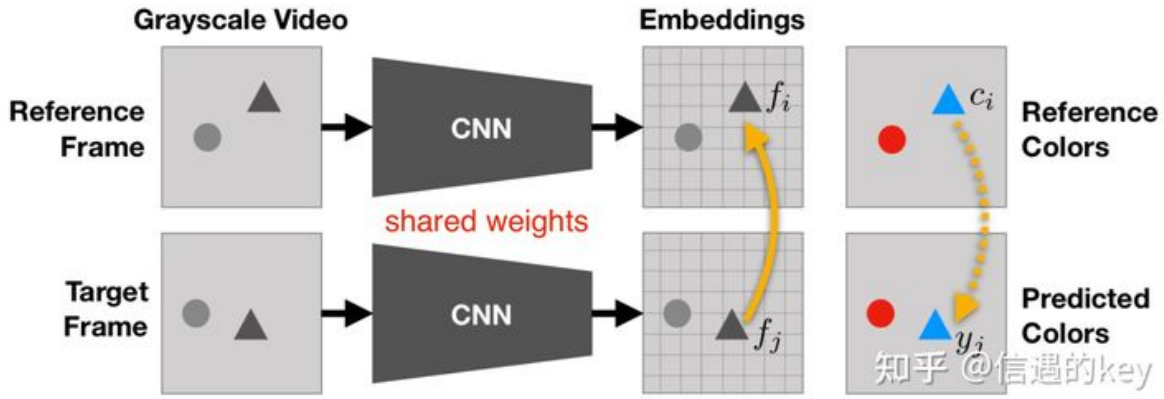
视频帧顺序识别：Shuffle and Learn: Unsupervised Learning using Temporal Order Verification

打乱输入帧的顺序，要求模型学习到正确的顺序。

视频上色（Video Colorization）

基于前一帧图像的颜色给后一帧图像上色。

> 与基于图像的上色不同，这里的任务是 **通过利用视频帧中颜色的自然时间相干性（因此这两个帧在时间上不应该相隔太远），将颜色从彩色的正常参考帧复制到灰度的另一个目标帧。**为了一致地复制颜色，该模型被设计成学习跟踪不同帧中的相关像素。



我的私货

学习更加通用的representation

在前面说过， 自监督学习的目的是在proxy task上训练，让模型学到数据的representation。 我们希望这个representation比原数据更简单，但包含原数据中全部的语义信息，这样才能在大多下游任务中表现优异。 以图片为例，我们希望通过定义的proxy task让模型学到图片中的纹理，形状，颜色等信息。最好的情况是模型能够学到原始图片中的所有信息，一个不落。 特定的任务需要特定的特征，我们只需要对学到的representation进行过滤，保留对该下游任务有用的特征即可。 这样学到的特征具有泛化性，就能用来处理不同的下游任务。

但由于proxy task的限制，模型被强迫关注与task相关的特征，而忽略其他数据中的语义信息。 因为这些语义信息与proxy task无关。 以预测颜色为例，我们希望模型更加关注图片中的颜色，因此纹理，形状等信息就会在预训练的过程中被忽略掉。 而这样学到的特征面对一些需要纹理或形状信息的下游任务就会表现很差。

现有的解决方案

1. 在SimCLR中，作者使用不同的augmentation以及不同augmentation 的组合，来保证每次生成样本都是包含不同的语义特征，防止模型只关注一种语义信息。
2. 在SimCLR的框架中，作者使用CNN-based network 作为encoder来捕获底层的语义信息h，在此之后使用Non-linear projection （MLP）对h做非线性变换，过滤掉h中与proxy task 无关的信息。 通过将模型分层两部分，前面部分用来捕获完整的语义信息，后一部分用来提炼task相关特征，防止损失函数强迫模型所有的层都要学习 pretext 相关的信息。 我个人认为由于损失函数的限制，前面的层或多或少会受到task的影响，忽略一些task无关的特征，只是程度多少的问题。

我的解决方案

模型的任务是由损失函数定义的，模型对特征的倾向也是受到损失函数影响。 既然如此，我们可以在同一个模型下定义多种损失函数，完成不同的任务。 其中不同的任务对数据特征的要求不同。 这样就能尽可能的保证所有的语义信息都能在训练过程中保留下来。 比如我们希望通过预训练让模型完成A,B,C 三个任务，这三个任务分别关注数据的颜色，形状，和纹理信息。 分别给这三个任务定义三个损失函数：a,b,c。 通过优化损失函数a,b,c， 这些信息都能在一次训练中被保留下来。

我感觉不错的proxy task

我做的题目是：通过历史数据预测（拟合）将来数据，需要考虑数据间的时序相关性。 以下的proxy task都与这个题目相关。

1. 上下文信息： 对于连续的历史数据，随机mask其中一个，让模型预测中被mask的数据。
2. predictive learning： 随机mask其中一个（一些）历史数据，要求模型预测出将来数据。
3. 乱序历史数据，预测出将来数据。
4. 将来数据预测过去。
5. augmentation： 使用随机初始化的卷积层对历史数据进行data augmentation。
6. augmentation：乱序（reorder）
7. augmentation： 高斯噪声

8. augmentation： Cutoff，dropout

Reference：

Contrastive Self-Supervised Learning | Ankesh Anand

Self-Supervised Learning 入门介绍 - 知乎 (zhihu.com)

对比学习（Contrastive Learning）:研究进展精要 - 知乎 (zhihu.com)

A Simple Framework for Contrastive Learning of Visual Representations

Understanding the Behaviour of Contrastive Loss

Self-Supervised Representation Learning (lilianweng.github.io)

自监督学习的一些思考 - 知乎 (zhihu.com)

基于视频的自监督学习 - 知乎 (zhihu.com)

计算机视觉 - 自监督学习 - Self-supervised Video Representation Learning*哔哩哔哩*bilibili