Invariant Information Clustering for Unsupervised Image Classification and Segmentation

Xu Ji
University of Oxford
xuji@robots.ox.ac.uk

João F. Henriques
University of Oxford
joao@robots.ox.ac.uk

Andrea Vedaldi
University of Oxford
vedaldi@robots.ox.ac.uk

Abstract

We present a novel clustering objective that learns a neural network classifier from scratch, given only unlabelled data samples. The model discovers clusters that accurately match semantic classes, achieving state-of-the-art results in eight unsupervised clustering benchmarks spanning image classification and segmentation. These include STL10, an unsupervised variant of ImageNet, and CIFAR10, where we significantly beat the accuracy of our closest competitors by 6.6 and 9.5 absolute percentage points respectively. The method is not specialised to computer vision and operates on any paired dataset samples; in our experiments we use random transforms to obtain a pair from each image. The trained network directly outputs semantic labels, rather than high dimensional representations that need external processing to be usable for semantic clustering. The objective is simply to maximise mutual information between the class assignments of each pair. It is easy to implement and rigorously grounded in information theory, meaning we effortlessly avoid degenerate solutions that other clustering methods are susceptible to. In addition to the fully unsupervised mode, we also test two semi-supervised settings. The first achieves 88.8% accuracy on STL10 classification, setting a new global state-of-the-art over all existing methods (whether supervised, semi-supervised or unsupervised). The second shows robustness to 90% reductions in label coverage, of relevance to applications that wish to make use of small amounts of labels. `github.com/xu-ji/IIC`

1. Introduction

Most supervised deep learning methods require large quantities of manually labelled data, limiting their applicability in many scenarios. This is true for large-scale image classification and even more for segmentation (pixel-wise classification) where the annotation cost per image is very high [38, 21]. Unsupervised clustering, on the other hand, aims to group data points into classes entirely without labels [25]. Many authors have sought to combine mature clustering algorithms with deep learning, for example by bootstrapping network training with k-means style objectives [51, 24, 7]. However, trivially combining clustering and representation learning methods often leads to degenerate solutions [7, 51]. It is precisely to prevent such degeneracy that cumbersome pipelines — involving pre-training, feature post-processing (whitening or PCA), clustering mechanisms external to the network — have evolved [7, 17, 18, 51].

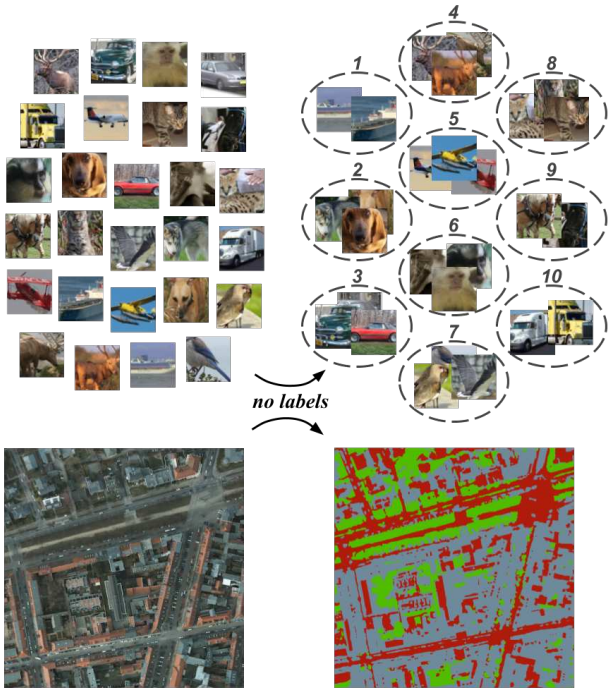

Figure 1: Models trained with IIC on entirely unlabelled data learn to cluster images (top, STL10) and patches (bottom, Potsdam-3). The raw clusters found directly correspond to semantic classes (dogs, cats, trucks, roads, vegetation etc.) with state-of-the-art accuracy. Training is end-to-end and randomly initialised, with no heuristics used at any stage.

In this paper, we introduce Invariant Information Clustering (IIC), a method that addresses this issue in a more principled manner. IIC is a generic clustering algorithm that

directly trains a randomly initialised neural network into a classification function, end-to-end and without any labels. It involves a simple objective function, which is the mutual information between the function's classifications for paired data samples. The input data can be of any modality and, since the clustering space is discrete, mutual information can be computed exactly.

Despite its simplicity, IIC is intrinsically robust to two issues that affect other methods. The first is clustering degeneracy, which is the tendency for a single cluster to dominate the predictions or for clusters to disappear (which can be observed with k-means, especially when combined with representation learning [7]). Due to the entropy maximisation component within mutual information, the loss is not minimised if all images are assigned to the same class. At the same time, it is optimal for the model to predict for each image a single class with certainty (i.e. one-hot) due to the conditional entropy minimisation (fig. 3). The second issue is noisy data with unknown or distractor classes (present in STL10 [10] for example). IIC addresses this issue by employing an auxiliary output layer that is parallel to the main output layer, trained to produce an overclustering (i.e. same loss function but greater number of clusters than the ground truth) that is ignored at test time. Auxiliary overclustering is a general technique that could be useful for other algorithms. These two features of IIC contribute to making it the only method amongst our unsupervised baselines that is robust enough to make use of the noisy unlabelled subset of STL10, a version of ImageNet [14] specifically designed as a benchmark for unsupervised clustering.

In the rest of the paper, we begin by explaining the difference between semantic clustering and intermediate representation learning (section 2), which separates our method from the majority of work in unsupervised deep learning. We then describe the theoretical foundations of IIC in statistical learning (section 3), demonstrating that maximising mutual information between pairs of samples under a bottleneck is a principled clustering objective which is equivalent to distilling their shared abstract content (co-clustering). We propose that for static images, an easy way to generate pairs with shared abstract content from unlabelled data is to take each image and its random transformation, or each patch and a neighbour. We show that maximising MI automatically avoids degenerate solutions and can be written as a convolution in the case of segmentation, allowing for efficient implementation with any deep learning library.

We perform experiments on a large number of datasets (section 4) including STL, CIFAR, MNIST, COCO-Stuff and Potsdam, setting a new state-of-the-art on unsupervised clustering and segmentation in all cases, with results of 59.6%, 61.7% and 72.3% on STL10, CIFAR10 and COCO-Stuff-3 beating the closest competitors (53.0%, 52.2%, 54.0%) with significant margins. Note that training deep neural networks to perform large scale, real-world segmentations from scratch, without labels or heuristics, is a highly challenging task with negligible precedent. We also perform an ablation study and additionally test two semi-supervised modes, setting a new global state-of-the-art of 88.8% on STL10 over all supervised, semi-supervised and unsupervised methods, and demonstrating the robustness in semi-supervised accuracy when 90% of labels are removed.

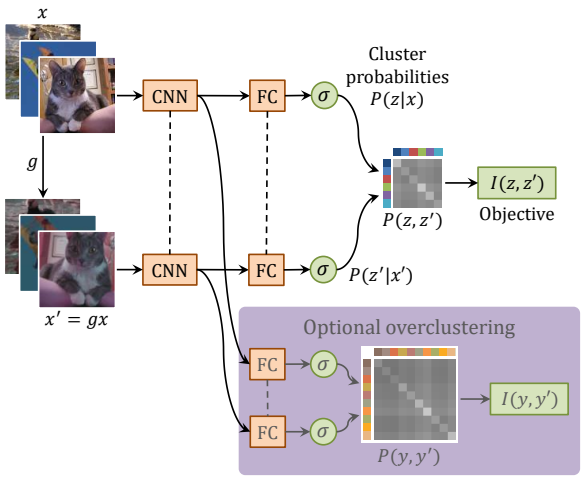

Figure 2: IIC for image clustering. Dashed line denotes shared parameters, g is a random transformation, and I denotes mutual information (eq. (3)).

2. Related work

Co-clustering and mutual information. The use of information as a criterion to learn representations is not new. One of the earliest works to do so is by Becker and Hinton [3]. More generally, learning from paired data has been explored in co-clustering [25, 16] and in other works [50] that build on the information bottleneck principle [20].

Several recent papers have used information as a tool to train deep networks in particular. IMSAT [28] maximises mutual information between data and its representation and DeepINFOMAX [27] maximizes information between spatially-preserved features and compact features. However, IMSAT and DeepINFOMAX combine information with other criteria, whereas in our method information is the only criterion used. Furthermore, both IMSAT and DeepINFOMAX compute mutual information over continuous random variables, which requires complex estimators [4], whereas IIC does so for discrete variables with simple and exact computations. Finally, DeepINFOMAX considers the information $I(\mathbf{x}, f(\mathbf{x}))$ between the features $\mathbf{x}$ and a deterministic function $f(\mathbf{x})$ of it, which is in principle the same as the entropy $H(\mathbf{x})$; in contrast, in IIC information does not trivially reduce to entropy.

Semantic clustering versus intermediate representation learning. In semantic clustering, the learned function directly outputs discrete assignments for high level (i.e.

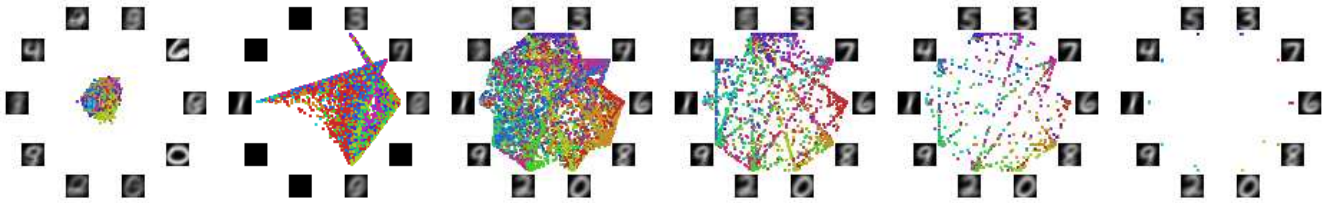

Figure 3: Training with IIC on unlabelled MNIST in successive epochs from random initialisation (left). The network directly outputs cluster assignment probabilities for input images, and each is rendered as a coordinate by convex combination of 10 cluster vertices. There is no cherry-picking as the entire dataset is shown in every snapshot. Ground truth labelling (unseen by model) is given by colour. At each cluster the average image of its assignees is shown. With neither labels nor heuristics, the clusters discovered by IIC correspond perfectly to unique digits, with one-hot certain prediction (right).

semantic) clusters. Intermediate representation learners, on the other hand, produce continuous, distributed, high-dimensional representations that must be post-processed, for example by k-means, to obtain the discrete low-cardinality assignments required for unsupervised semantic clustering. The latter includes objectives such as generative autoencoder image reconstruction [48], triplets [46] and spatial-temporal order or context prediction [37, 12, 17], for example predicting patch proximity [30], solving jigsaw puzzles [41] and inpainting [43]. Note it also includes a number of clustering methods (DeepCluster [7], exemplars [18]) where the clustering is only auxiliary; a clustering-style objective is used but does not produce groups with semantic correspondence. For example, DeepCluster [7] is a state-of-the-art method for learning highly-transferable intermediate features using overclustering as a proxy task, but does not automatically find semantically meaningful clusters. As these methods use auxiliary objectives divorced from the semantic clustering objective, it is unsurprising that they perform worse than IIC (section 4), which directly optimises for it, training the network end-to-end with the final clusterer implicitly wrapped inside.

Optimising image-to-image distance. Many approaches to deep clustering, whether semantic or auxiliary, utilise a distance function between input images that approximates a given grouping criterion. Agglomerative clustering [2] and partially ordered sets [1] of HOG features [13] have been used to group images, and exemplars [18] define a group as a set of random transformations applied to a single image. Note the latter does not scale easily, in particular to image segmentation where a single 200×200 image would call for 40k classes. DAC [8], JULE [52], DeepCluster [7], ADC [24] and DEC [51] rely on the inherent visual consistency and disentangling properties [23] of CNNs to produce cluster assignments, which are processed and reinforced in each iteration. The latter three are based on k-means style mechanisms to refine feature centroids, which is prone to degenerate solutions [7] and thus needs explicit prevention mechanisms such as pre-training, cluster-reassignment or feature cleaning via PCA and whitening [51, 7].

Invariance as a training objective. Optimising for function outputs to be persistent through spatio-temporal or non-material distortion is an idea shared by IIC with several works, including exemplars [18], IMSAT [28], proximity prediction [30], the denoising objective of Tagger [22], temporal slowness constraints [55], and optimising for features to be invariant to local image transformations [47, 29]. More broadly, the problem of modelling data transformation has received significant attention in deep learning, one example being the transforming autoencoder [26].

3. Method

First we introduce a generic objective, Invariant Information Clustering, which can be used to cluster any kind of unlabelled paired data by training a network to predict cluster identities (section 3.1). We then apply it to image clustering (section 3.2, fig. 2 and fig. 3) and segmentation (section 3.3), by generating the required paired data using random transformations and spatial proximity.

3.1. Invariant Information Clustering

Let $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ be a paired data sample from a joint probability distribution $P(\mathbf{x}, \mathbf{x}')$. For example, $\mathbf{x}$ and $\mathbf{x}'$ could be different images containing the same object. The goal of Invariant Information Clustering (IIC) is to learn a representation $\Phi : \mathcal{X} \rightarrow \mathcal{Y}$ that preserves what is in common between $\mathbf{x}$ and $\mathbf{x}'$ while discarding instance-specific details. The former can be achieved by maximizing the mutual information between encoded variables:

$$\max_{\Phi} I(\Phi(\mathbf{x}), \Phi(\mathbf{x}')), \tag{1}$$

which is equivalent to maximising the predictability of $\Phi(\mathbf{x})$ from $\Phi(\mathbf{x}')$ and vice versa.

An effect of equation eq. (1), in general, is to make representations of paired samples the same. However, it is not the same as merely minimising representation distance, as done for example in methods based on k-means [7, 24]: the presence of entropy within I allows us to avoid degeneracy, as discussed in detail below.

If Φ is a neural network with a small output capacity (often called a "bottleneck"), eq. (1) also has the effect of discarding instance-specific details from the data. Clustering imposes a natural bottleneck, since the representation

space is $\mathcal{Y} = \{1, \ldots, C\}$, a finite set of class indices (as opposed to an infinite vector space). Without a bottleneck, i.e. assuming unbounded capacity, eq. (1) is trivially solved by setting Φ to the identity function because of the data processing inequality [11], i.e. $I(\mathbf{x}, \mathbf{x}') \geq I(\Phi(\mathbf{x}), \Phi(\mathbf{x}'))$.

Since our goal is to learn the representation with a deep neural network, we consider soft rather than hard clustering, meaning the neural network Φ is terminated by a (differentiable) softmax layer. Then the output $\Phi(\mathbf{x}) \in [0, 1]^C$ can be interpreted as the distribution of a discrete random variable z over C classes, formally given by $P(z = c|\mathbf{x}) = \Phi_c(\mathbf{x})$. Making the output probabilistic amounts to allowing for uncertainty in the cluster assigned to an input.

Consider now a pair of such cluster assignment variables z and z' for two inputs $\mathbf{x}$ and $\mathbf{x}'$ respectively. Their conditional joint distribution is given by $P(z = c, z' = c'|\mathbf{x}, \mathbf{x}') = \Phi_c(\mathbf{x}) \cdot \Phi_{c'}(\mathbf{x}')$. This equation states that z and z' are independent when conditioned on specific inputs $\mathbf{x}$ and $\mathbf{x}'$; however, in general they are *not* independent after marginalization over a dataset of input pairs $(\mathbf{x}_i, \mathbf{x}'_i)$, $i = 1, \ldots, n$. For example, for a trained classification network Φ and a dataset of image pairs where each image contains the same object of its pair but in a randomly different position, the random variable constituted by the class of the first of each pair, z, will have a strong statistical relationship with the random variable for the class of the second of each pair, z'; one is predictive of the other (in fact identical to it, in this case) so they are highly dependent. After marginalization over the dataset (or batch, in practice), the joint probability distribution is given by the $C \times C$ matrix $\mathbf{P}$, where each element at row c and column c' constitutes $\mathbf{P}_{cc'} = P(z = c, z' = c')$:

$$\mathbf{P} = \frac{1}{n} \sum_{i=1}^{n} \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}'_i)^\top. \tag{2}$$

The marginals $\mathbf{P}_c = P(z = c)$ and $\mathbf{P}_{c'} = P(z' = c')$ can be obtained by summing over the rows and columns of this matrix. As we generally consider symmetric problems, where for each $(\mathbf{x}_i, \mathbf{x}'_i)$ we also have $(\mathbf{x}'_i, \mathbf{x}_i)$, $\mathbf{P}$ is symmetrized using $(\mathbf{P} + \mathbf{P}^\top)/2$.

Now the objective function eq. (1) can be computed by plugging the matrix $\mathbf{P}$ into the expression for mutual information [36], which results in the formula:

$$I(z, z') = I(\mathbf{P}) = \sum_{c=1}^{C} \sum_{c'=1}^{C} \mathbf{P}_{cc'} \cdot \ln \frac{\mathbf{P}_{cc'}}{\mathbf{P}_c \cdot \mathbf{P}_{c'}}. \tag{3}$$

Why degenerate solutions are avoided. Mutual information (3) expands to $I(z, z') = H(z) - H(z|z')$. Hence, maximizing this quantity trades-off minimizing the conditional cluster assignment entropy $H(z|z')$ and maximising individual cluster assignments entropy $H(z)$. The smallest value of $H(z|z')$ is 0, obtained when the cluster assignments are exactly predictable from each other. The largest value of $H(z)$ is $\ln C$, obtained when all clusters are equally likely to be picked. This occurs when the data is assigned evenly between the clusters, equalizing their mass. Therefore the loss is not minimised if all samples are assigned to a single cluster (i.e. output class is identical for all samples). Thus as maximising mutual information naturally balances reinforcement of predictions with mass equalization, it avoids the tendency for degenerate solutions that algorithms which combine k-means with representation learning are susceptible to [7]. For further discussion of entropy maximisation, and optionally how to prioritise it with an entropy coefficient, see supplementary material.

Meaning of mutual information. The reader may now wonder what are the benefits of maximising mutual information, as opposed to merely maximising entropy. Firstly, due to the soft clustering, entropy alone could be maximised trivially by setting all prediction vectors $\Phi(\mathbf{x})$ to uniform distributions, resulting in no clustering. This is corrected by the conditional entropy component, which encourages deterministic one-hot predictions. For example, even for the degenerate case of identical pairs $\mathbf{x} = \mathbf{x}'$, the IIC objective encourages a deterministic clustering function (i.e. $\Phi(\mathbf{x})$ is a one-hot vector) as this results in null conditional entropy $H(z|z') = 0$. Secondly, the objective of IIC is to find what is common between two data points that share redundancy, such as different images of the same object, explicitly encouraging distillation of the common part while ignoring the rest, i.e. instance details specific to one of the samples. This would not be possible without pairing samples.

3.2. Image clustering

IIC requires a source of paired samples $(\mathbf{x}, \mathbf{x}')$, which are often unavailable in unsupervised image clustering applications. In this case, we propose to use *generated* image pairs, consisting of image $\mathbf{x}$ and its *randomly perturbed* version $\mathbf{x}' = g\mathbf{x}$. The objective eq. (1) can thus be written as:

$$\max_{\Phi} I(\Phi(\mathbf{x}), \Phi(g\mathbf{x})), \tag{4}$$

where both image $\mathbf{x}$ and transformation g are random variables. Useful g could include scaling, skewing, rotation or flipping (geometric), changing contrast and colour saturation (photometric), or any other perturbation that is likely to leave the content of the image intact. IIC can then be used to recover the factor which is *invariant* to which of the pair is picked. The effect is to learn a function that partitions the data such that clusters are closed to the perturbations, without dropping clusters. The objective is simple enough to be written in six lines of PyTorch code (fig. 4).

Auxiliary overclustering. For certain datasets (e.g. STL10), training data comes in two types: one known to contain only relevant classes and the other known to contain irrelevant or distractor classes. It is desirable to train a

```
def IIC(z, zt, C=10):
  P = (z.unsqueeze(2) * zt.unsqueeze(1)).sum(dim=0)
  P = ((P + P.t()) / 2) / P.sum()
  P[(P < EPS).data] = EPS
  Pi = P.sum(dim=1).view(C, 1).expand(C, C)
  Pj = P.sum(dim=0).view(1, C).expand(C, C)
  return (P * (log(Pi) + log(Pj) - log(P))).sum()
```

Figure 4: IIC objective in PyTorch. Inputs `z` and `zt` are $n \times C$ matrices, with C predicted cluster probabilities for n sampled pairs (i.e. CNN softmaxed predictions). For example, the prediction for each image in a dataset and its transformed version (e.g. using standard data augmentation).

clusterer specialised for the relevant classes, that still benefits from the context provided by the distractor classes, since the latter is often much larger (for example 100K compared to 13K for STL10). Our solution is to add an auxiliary overclustering head to the network (fig. 2) that is trained with the full dataset, whilst the main output head is trained with the subset containing only relevant classes. This allows us to make use of the noisy unlabelled subset despite being an unsupervised clustering method. Other methods are generally not robust enough to do so and thus avoid the 100k-samples unlabelled subset of STL10 when training for unsupervised clustering ([8, 24, 51]). Since the auxiliary overclustering head outputs predictions over a larger number of clusters than the ground truth, whilst still maintaining a predictor that is matched to ground truth number of clusters (the main head), it can be useful in general for increasing expressivity in the learned feature representation, even for datasets where there are no distractor classes [7].

3.3. Image segmentation

IIC can be applied to image segmentation identically to image clustering, except for two modifications. Firstly, since predictions are made for each pixel densely, clustering is applied to image patches (defined by the receptive field of the neural network for each output pixel) rather than whole images. Secondly, unlike with whole images, one has access to the spatial relationships between patches. Thus, we can add *local spatial invariance* to the list of geometric and photometric invariances in section 3.2, meaning we form pairs of patches not only via synthetic perturbations, but also by extracting pairs of adjacent patches in the image.

In detail, let the RGB image $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ be a tensor, $u \in \Omega = \{1,\dots,H\} \times \{1,\dots,W\}$ a pixel location, and $\mathbf{x}_u$ a patch centered at u. We can form a pair of patches $(\mathbf{x}_u, \mathbf{x}_{u+t})$ by looking at location u and its neighbour $u+t$ at some small displacement $t \in T \subset \mathbb{Z}^2$. The cluster probability vectors for all patches $\mathbf{x}_u$ can be read off as the column vectors $\Phi(\mathbf{x}_u) = \Phi_u(\mathbf{x}) \in [0,1]^C$ of the tensor $\Phi(\mathbf{x}) \in [0,1]^{C\times H\times W}$, computed by a single application of the convolutional network Φ. Then, to apply IIC, one simply substitutes pairs $(\Phi_u(\mathbf{x}), \Phi_{u+t}(\mathbf{x}))$ in the calculation of the joint probability matrix (2).

The geometric and photometric perturbations used before for whole image clustering can be applied to individual patches too. Rather than transforming patches individually, however, it is much more efficient to transform all of them in parallel by perturbing the entire image. Any number or combination of these invariances can be chained and learned simultaneously; the only detail is to ensure indices of the original image and transformed image class probability tensors line up, meaning that predictions from patches which are intended to be paired together do so.

Formally, if the image transformation g is a geometric transformation, the vector of cluster probabilities $\Phi_u(\mathbf{x})$ will not correspond to $\Phi_u(g\mathbf{x})$; rather, it will correspond to $\Phi_{g(u)}(g\mathbf{x})$ because patch $\mathbf{x}_u$ is sent to patch $\mathbf{x}_{g(u)}$ by the transformation. All vectors can be paired at once by applying the reverse transformation g^{-1} to the tensor $\Phi(g\mathbf{x})$, as $[g^{-1}\Phi(g\mathbf{x})]_u = \Phi_{g(u)}(g\mathbf{x})$. For example, flipping the input image will require flipping the resulting probability tensor back. In general, the perturbation g can incorporate geometric and photometric transformations, and g^{-1} only needs to undo geometric ones. The segmentation objective is thus:

$$\max_{\Phi} \frac{1}{|T|}\sum_{t\in T} I(\mathbf{P}_t), \qquad (5)$$
$$\mathbf{P}_t = \frac{1}{n|G||\Omega|}\sum_{i=1}^{n}\sum_{g\in G}\overbrace{\sum_{u\in\Omega}\Phi_u(\mathbf{x}_i)\cdot[g^{-1}\Phi(g\mathbf{x}_i)]_{u+t}^{\top}}^{\text{Convolution}}.$$

Hence the goal is to maximize the information between each patch label $\Phi_u(\mathbf{x}_i)$ and the patch label $[g^{-1}\Phi(g\mathbf{x}_i)]_{u+t}$ of its transformed neighbour patch, in expectation over images $i = 1,\dots,n$, patches $u \in \Omega$ within each image, and perturbations $g \in G$. Information is in turn averaged over all neighbour displacements $t \in T$ (which was found to perform slightly better than averaging over t before computing information; see supplementary material).

Implementation. The joint distribution of eq. (5) for all displacements $t \in T$ can be computed in a simple and highly efficient way. Given two network outputs for one batch of image pairs $\mathbf{y} = \Phi(\mathbf{x}), \mathbf{y}' = \Phi(g\mathbf{x})$ where $\mathbf{y}, \mathbf{y}' \in \mathbb{R}^{n\times C\times H\times W}$, we first bring $\mathbf{y}'$ back into the coordinate-space of $\mathbf{y}$ by using a bilinear resampler[1] [32], which inverts any geometrical transforms in g, $\mathbf{y}' \leftarrow g^{-1}\mathbf{y}'$. Then, the inner summation in eq. (5) reduces to the convolution of the two tensors. Using any standard deep learning framework, this can be achieved by swapping the first two dimensions of each of y and y', computing $\mathbf{P} = y * y'$ (a 2D convolution with padding d in both dimensions), and normalising the result to produce $\mathbf{P} \in [0,1]^{C\times C\times(2d+1)\times(2d+1)}$.

4. Experiments

We apply IIC to fully unsupervised image clustering and segmentation, as well as two semi-supervised settings. Ex-

[1] The core differentiable operator in spatial transformer networks [32].

	STL10	CIFAR10	CFR100-20	MNIST
Random network	13.5	13.1	5.93	26.1
K-means [53]†	19.2	22.9	13.0	57.2
Spectral clustering [49]	15.9	24.7	13.6	69.6
Triplets [46]‡	24.4	20.5	9.94	52.5
AE [5]‡	30.3	31.4	16.5	81.2
Sparse AE [40]‡	32.0	29.7	15.7	82.7
Denoising AE [48]‡	30.2	29.7	15.1	83.2
Variational Bayes AE [34]‡	28.2	29.1	15.2	83.2
SWWAE 2015 [54]‡	27.0	28.4	14.7	82.5
GAN 2015 [45]‡	29.8	31.5	15.1	82.8
JULE 2016 [52]	27.7	27.2	13.7	96.4
DEC 2016 [51]†	35.9	30.1	18.5	84.3
DAC 2017 [8]	47.0	52.2	23.8	97.8
DeepCluster 2018 [7]† ‡	33.4⋆	37.4⋆	18.9⋆	65.6 ⋆
ADC 2018 [24]	53.0	32.5	16.0⋆	99.2
IIC (lowest loss sub-head)	**59.6**	**61.7**	**25.7**	**99.2**
IIC (avg sub-head ± STD)	59.8 ± 0.844	57.6 ± 5.01	25.5 ± 0.462	98.4 ± 0.652

Table 1: **Unsupervised image clustering.** Legend: †Method based on k-means. ‡Method that does not directly learn a clustering function and requires further application of k-means to be used for image clustering. ⋆Results obtained using our experiments with authors' original code.

	STL10
No auxiliary overclustering	43.8
Single sub-head ($h = 1$)	57.6
No sample repeats ($r = 1$)	47.0
Unlabelled data segment ignored	49.9
Full setting	**59.6**

Table 2: **Ablations of IIC (unsupervised setting).** Each row shows a single change from the full setting. The full setting has auxiliary overclustering, 5 initialisation heads, 5 sample repeats, and uses the unlabelled data subset of STL10.

isting baselines are outperformed in all cases. We also conduct an analysis of our method via ablation studies. For minor details see supplementary material.

4.1. Image clustering

Datasets. We test on STL10, which is ImageNet adapted for unsupervised classification, as well as CIFAR10, CIFAR100-20 and MNIST. The main setting is pure unsupervised clustering (IIC) but we also test two semi-supervised settings: *finetuning* and *overclustering*. For unsupervised clustering, following previous work [8, 51, 52], we train on the full dataset and test on the labelled part; for the semi-supervised settings, train and test sets are separate.

As for DeepCluster [7], we found Sobel filtering to be beneficial, as it discourages clustering based on trivial cues such as colour and encourages using more meaningful cues such as shape. Additionally, for data augmentation, we repeat images within each batch r times; this means that multiple image pairs within a batch contain the same original image, each paired with a different transformation, which encourages greater distillation since there are more examples of which visual details to ignore (section 3.1). We set $r \in [1, 5]$ for all experiments. Images are rescaled and cropped for training (prior to applying transforms g, consisting of random additive and multiplicative colour transformations and horizontal flipping) and a single center crop is used at test time for all experiments except semi-supervised finetuning, where 10 crops are used.

Architecture. All networks are randomly initialised and consist of a ResNet or VGG11-like base b (see sup. mat.), followed by one or more heads (linear predictors). Let the number of ground truth clusters be k_{gt} and the output channels of a head be k. For IIC, there is a main output head with $k = k_{gt}$ and an auxiliary overclustering head (fig. 2) with $k > k_{gt}$. For semi-supervised overclustering there is one output head with $k > k_{gt}$. For increased robustness, each head is duplicated $h = 5$ times with a different random initialisation, and we call these concrete instantiations sub-heads. Each sub-head takes features from b and outputs a probability distribution for each batch element over the relevant number of clusters. For semi-supervised finetuning (table 3), the base is copied from a semi-supervised overclustering network and combined with a single randomly initialised linear layer where $k = k_{gt}$.

Training. We use the Adam optimiser [33] with learning rate 10^{-4}. For IIC, the main and auxiliary heads are trained by maximising eq. (3) in alternate epochs. For semi-supervised overclustering, the single head is trained by maximising eq. (3). Semi-supervised finetuning uses a standard logistic loss.

Evaluation. We evaluate based on accuracy (true positives divided by sample size). For IIC we follow the standard protocol of finding the best one-to-one permutation mapping between learned and ground-truth clusters (from the main output head; auxiliary overclustering head is ignored) using linear assignment [35]. While this step uses labels, it does not constitute learning as it merely makes the metric invariant to the order of the clusters. For semi-supervised overclustering, each ground-truth cluster may correspond to the union of several predicted clusters. Evaluation thus requires a many-to-one discrete map from k to k_{gt}, since $k > k_{gt}$. This extracts some information from the labels and thus requires separated training and test set. Note this mapping is found using the training set (accuracy is computed on the test set) and does not affect the network parameters as it is used for evaluation only. For semi-supervised finetuning, output channel order matches ground truth so no mapping is required. Each sub-head is assessed independently; we report average and best sub-head (as chosen by lowest IIC loss) performance.

Unsupervised learning analysis. IIC is highly capable of discovering clusters in unlabelled data that accurately correspond to the underlying semantic classes, and outperforms all competing baselines at this task (table 1), with significant margins of 6.6% and 9.5% in the case of STL10 and CI-

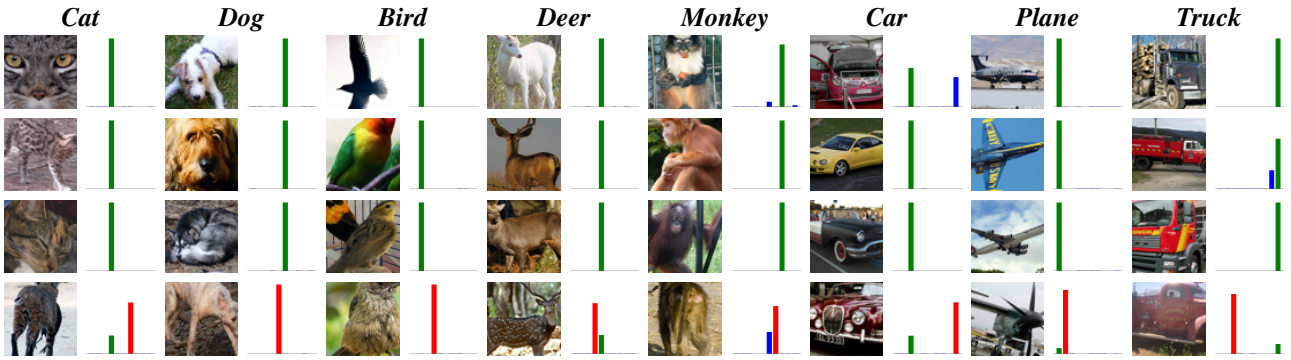

Figure 5: **Unsupervised image clustering (IIC) results on STL10.** Predicted cluster probabilities from the best performing head are shown as bars. Prediction corresponds to tallest, ground truth is green, incorrectly predicted classes are red, and all others are blue. The bottom row shows failure cases.

	STL10
Dosovitskiy 2015 [18]†	74.2
SWWAE 2015 [54]†	74.3
Dundar 2015 [19]	74.1
Cutout* 2017 [15]	87.3
Oyallon* 2017 [42]†	76.0
Oyallon* 2017 [42]	87.6
DeepCluster 2018 [7]	73.4⋆
ADC 2018 [24]	56.7⋆
DeepINFOMAX 2018 [27]	77.0
IIC plus finetune†	**79.2**
IIC plus finetune	**88.8**

Table 3: **Fully and semi-supervised classification.** Legend: *Fully supervised method. ⋆Our experiments with authors' code. †Multi-fold evaluation.

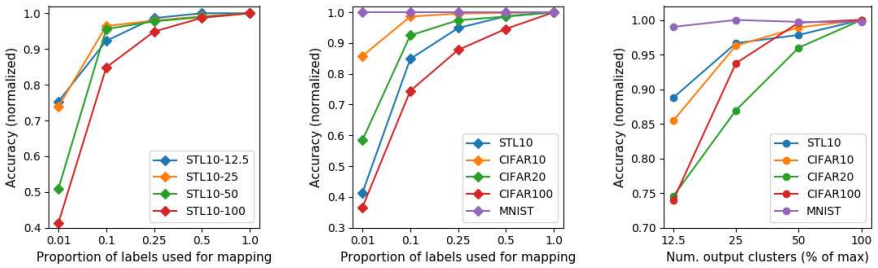

Figure 6: **Semi-supervised overclustering.** Training with IIC loss to overcluster ($k > k_{gt}$) and using labels for evaluation mapping only. Performance is robust even with 90%-75% of labels discarded (left and center). STL10-r denotes networks with output $k = \lceil 1.4r \rceil$. Overall accuracy improves with the number of output clusters k (right). For further details see supplementary material.

FAR10. As mentioned in section 2, this underlines the advantages of end-to-end optimisation instead of using a fixed external procedure like k-means as with many baselines. The clusters found by IIC are highly discriminative (fig. 5), although note some failure cases; as IIC distills purely visual correspondences within images, it can be confused by instances that combine classes, such as a deer with the coat pattern of a cat. Our ablations (table 2) illustrate the contributions of various implementation details, and in particular the accuracy gain from using auxiliary overclustering.

Semi-supervised learning analysis. For semi-supervised learning, we establish a new state-of-the-art on STL10 out of all reported methods by finetuning a network trained in an entirely unsupervised fashion with the IIC objective (recall labels in semi-supervised overclustering are used for evaluation and do not influence the network parameters). This explicitly validates the quality of our unsupervised learning method, as we beat even the supervised state-of-the-art (table 3). Given that the bulk of parameters within semi-supervised overclustering are trained unsupervised (i.e. all network parameters), it is unsurprising that Figure 6 shows a 90% drop in the number of available labels for STL10 (decreasing the amount of labelled data available from 5000 to 500 over 10 classes) barely impacts performance, costing just ∼10% drop in accuracy. This setting has lower label requirements than finetuning because whereas the latter learns all network parameters, the former only needs to learn a discrete map between k and k_{gt}, making it an important practical setting for applications with small amounts of labelled data.

4.2. Segmentation

Datasets. Large scale segmentation on real-world data using deep neural networks is extremely difficult without labels or heuristics, and has negligible precedent. We establish new baselines on scene and satellite images to highlight performance on textural classes, where the assumption of spatially proximal invariance (section 3.3) is most valid. COCO-Stuff [6] is a challenging and diverse segmentation dataset containing "stuff" classes ranging from buildings to bodies of water. We use the 15 coarse labels and 164k images variant, reduced to 52k by taking only images with at least 75% stuff pixels. COCO-Stuff-3 is a subset of COCO-Stuff with only sky, ground and plants labelled. For both COCO datasets, input images are shrunk by two thirds and cropped to 128 × 128 pixels, Sobel preprocessing is applied for data augmentation, and predictions for non-stuff pixels are ignored. Potsdam [31] is divided into 8550 RGBIR 200 × 200 px satellite images, of which 3150 are unlabelled. We test both the 6-label variant (roads and cars, vegetation and trees, buildings and clutter) and a 3-label variant (Potsdam-3) formed by merging each of the 3 pairs. All segmentation training and testing sets have been released with our code.

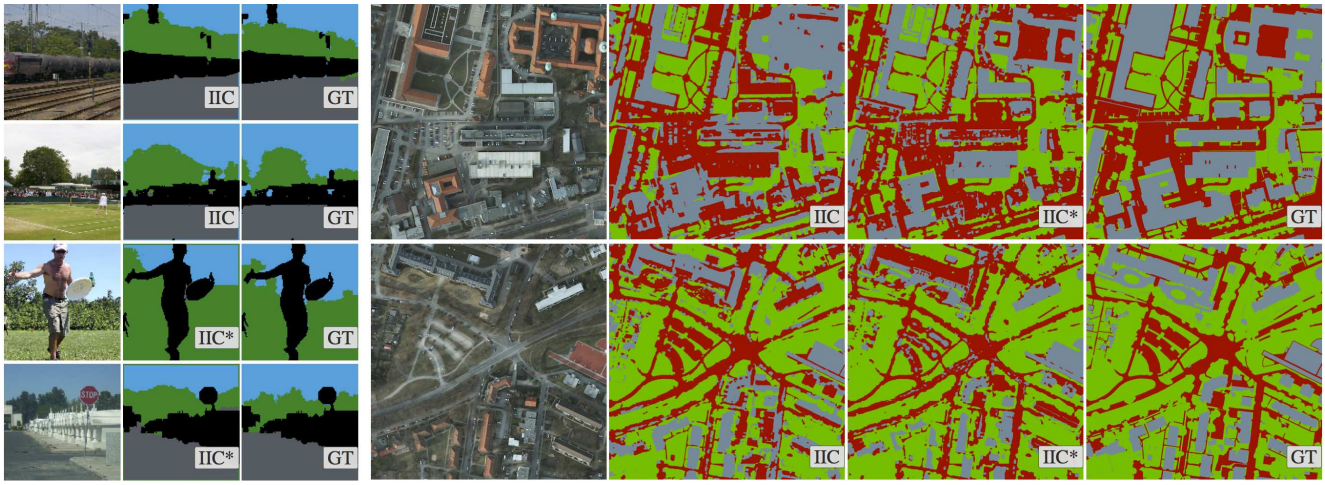

Figure 7: **Example segmentation results (un- and semi-supervised).** Left: COCO-Stuff-3 (non-stuff pixels in black), right: Potsdam-3. Input images, IIC (fully unsupervised segmentation) and IIC* (semi-supervised overclustering) results are shown, together with the ground truth segmentation (GT).

	COCO-Stuff-3	COCO-Stuff	Potsdam-3	Potsdam
Random CNN	37.3	19.4	38.2	28.3
K-means [44]†	52.2	14.1	45.7	35.3
SIFT [39]‡	38.1	20.2	38.2	28.5
Doersch 2015 [17]‡	47.5	23.1	49.6	37.2
Isola 2016 [30]‡	54.0	24.3	63.9	44.9
DeepCluster 2018 [7]† ‡	41.6	19.9	41.7	29.2
IIC	**72.3**	**27.7**	**65.1**	**45.4**

Table 4: **Unsupervised segmentation.** IIC experiments use a single subhead. Legend: †Method based on k-means. ‡Method that does not directly learn a clustering function and requires further application of k-means to be used for image clustering.

Architecture. All networks are randomly initialised and consist of a base CNN b (see sup. mat.) followed by head(s), which are 1×1 convolution layers. Similar to section 4.1, overclustering uses k 3-5 times higher than k_{gt}. Since segmentation is much more expensive than image clustering (e.g. a single 200×200 Potsdam image contains 40,000 predictions), all segmentation experiments were run with $h = 1$ and $r = 1$ (sec. 4.1).

Training. The convolutional implementation of IIC (eq. (5)) was used with $d = 10$. For Potsdam-3 and COCO-Stuff-3, the optional entropy coefficient (section 3.1 and sup. mat.) was used and set to 1.5. Using the coefficient made slight improvements of 1.2%-3.2% on performance. These two datasets are balanced in nature with very large sample volume (e.g. $40,000 \times 75$ predictions per batch for Potsdam-3) resulting in stable and balanced batches, justifying prioritisation of equalisation. Other training details are the same as section 4.1.

Evaluation. Evaluation uses accuracy as in section 4.1, computed per-pixel. For the baselines, the original authors' code was adapted from image clustering where available, and the architectures are shared with IIC for fairness. For baselines that required application of k-means to produce per-pixel predictions (table 4), k-means was trained with randomly sampled pixel features from the training set (10M for Potsdam, Potsdam-3; 50M for COCO-Stuff, COCO-Stuff-3) and tested on the full test set to obtain accuracy.

Analysis. Without labels or heuristics to learn from, and given just the cluster cardinality (3), IIC automatically partitions COCO-Stuff-3 into clusters that are recognisable as sky, vegetation and ground, and learns to classify vegetation, roads and buildings for Potsdam-3 (fig. 7). The segmentations are notably intricate, capturing fine detail, but are at the same time locally consistent and coherent across all images. Since spatial smoothness is built into the loss (section 3.3), all our results are able to use raw network outputs without post-processing (avoiding e.g. CRF smoothing [9]). Quantitatively, we outperform all baselines (table 4), notably by 18.3% in the case of COCO-Stuff-3. The efficient convolutional formulation of the loss (eq. (5)) allows us to optimise over all pixels in all batch images in parallel, converging in fewer epochs (passes of the dataset) without paying the price of reduced computational speed for dense sampling. This is in contrast to our baselines which, being not natively adapted for segmentation, required sampling a subset of pixels within each batch, resulting in increased loss volatility and training speeds that were up to 3.3× slower than IIC.

5. Conclusions

We have shown that it is possible to train neural networks into semantic clusterers without using labels or heuristics. The novel objective presented relies on statistical learning, by optimising mutual information between related pairs - a relationship that can be generated by random transforms - and naturally avoids degenerate solutions. The resulting models classify and segment images with state-of-the-art levels of semantic accuracy. Being not specific to vision, the method opens up many interesting research directions, including optimising information in datastreams over time.

Acknowledgments. We are grateful to ERC StG IDIU-638009 and EPSRC AIMS CDT for support.

References

[1] Miguel A Bautista, Artsiom Sanakoyeu, and Bjorn Ommer. Deep unsupervised similarity learning using partially ordered sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7130–7139, 2017. 3

[2] Miguel A Bautista, Artsiom Sanakoyeu, Ekaterina Tikhoncheva, and Bjorn Ommer. Cliquecnn: Deep unsupervised exemplar learning. In *Advances in Neural Information Processing Systems*, pages 3846–3854, 2016. 3

[3] Suzanna Becker and Geoffrey E Hinton. Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355(6356):161, 1992. 2

[4] Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R Devon Hjelm, and Aaron Courville. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018. 2

[5] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007. 6

[6] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. *arXiv preprint arXiv:1612.03716*, 2016. 7

[7] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. *arXiv preprint arXiv:1807.05520*, 2018. 1, 2, 3, 4, 5, 6, 7, 8

[8] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5879–5887, 2017. 3, 5, 6

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018. 8

[10] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 2

[11] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012. 4

[12] Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. Deeppermnet: Visual permutation learning. *arXiv preprint arXiv:1704.02729*, 2017. 3

[13] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005. 3

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[15] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 7

[16] Inderjit S Dhillon, Subramanyam Mallela, and Dharmendra S Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2003. 2

[17] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 1, 3, 8

[18] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015. 1, 3, 7

[19] Aysegul Dundar, Jonghoon Jin, and Eugenio Culurciello. Convolutional clustering for unsupervised learning. *arXiv preprint arXiv:1511.06241*, 2015. 7

[20] Nir Friedman, Ori Mosenzon, Noam Slonim, and Naftali Tishby. Multivariate information bottleneck. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 152–161. Morgan Kaufmann Publishers Inc., 2001. 2

[21] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 1

[22] Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hao, Harri Valpola, and Juergen Schmidhuber. Tagger: Deep unsupervised perceptual grouping. In *Advances in Neural Information Processing Systems*, pages 4484–4492, 2016. 3

[23] Klaus Greff, Rupesh Kumar Srivastava, and Jürgen Schmidhuber. Binding via reconstruction clustering. *arXiv preprint arXiv:1511.06418*, 2015. 3

[24] Philip Haeusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. Associative deep clustering: Training a classification network with no labels. In *German Conference on Pattern Recognition*, pages 18–32. Springer, 2018. 1, 3, 5, 6, 7

[25] John A Hartigan. Direct clustering of a data matrix. *Journal of the american statistical association*, 67(337):123–129, 1972. 1, 2

[26] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011. 3

[27] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 2, 7

[28] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. *arXiv preprint arXiv:1702.08720*, 2017. 2, 3

[29] Ka Yu Hui. Direct modeling of complex invariances for visual object features. In *International Conference on Machine Learning*, pages 352–360, 2013. 3

[30] Phillip Isola, Daniel Zoran, Dilip Krishnan, and Edward H Adelson. Learning visual groups from co-occurrences in space and time. *arXiv preprint arXiv:1511.06811*, 2015. 3, 8

[31] ISPRS. ISPRS 2D Semantic Labeling Contest. http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html. [Online; accessed 10-October-2018]. 7

[32] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 5

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 6

[35] Harold W Kuhn. The hungarian method for the assignment problem. In *50 Years of Integer Programming 1958-2008*, pages 29–47. Springer, 2010. 6

[36] Erik G Learned-Miller. Entropy and mutual information. 4

[37] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 667–676. IEEE, 2017. 3

[38] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proc. ECCV*, pages 740–755. Springer, 2014. 1

[39] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 8

[40] Andrew Ng. Sparse autoencoder. *CS294A Lecture notes*, pages 1–19, 2011. 6

[41] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 3

[42] Edouard Oyallon, Eugene Belilovsky, and Sergey Zagoruyko. Scaling the scattering transform: Deep hybrid networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5618–5627, 2017. 7

[43] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016. 3

[44] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011. 8

[45] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 6

[46] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *Advances in neural information processing systems*, pages 41–48, 2004. 3, 6

[47] Kihyuk Sohn and Honglak Lee. Learning invariant representations with local transformations. *arXiv preprint arXiv:1206.6418*, 2012. 3

[48] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010. 3, 6

[49] Jianfeng Wang, Jingdong Wang, Jingkuan Song, Xin-Shun Xu, Heng Tao Shen, and Shipeng Li. Optimized cartesian k-means. *IEEE Transactions on Knowledge & Data Engineering*, (1):1–1, 2015. 6

[50] Pu Wang, Carlotta Domeniconi, and Kathryn Blackmond Laskey. Information bottleneck co-clustering. Citeseer. 2

[51] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016. 1, 3, 5, 6

[52] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5147–5156, 2016. 3, 6

[53] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2005. 6

[54] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. Stacked what-where auto-encoders. *arXiv preprint arXiv:1506.02351*, 2015. 6, 7

[55] Will Zou, Shenghuo Zhu, Kai Yu, and Andrew Y Ng. Deep learning of invariant features via simulated fixations in video. In *Advances in neural information processing systems*, pages 3203–3211, 2012. 3