курс «Глубокое обучение»

Самообучение (Self-Supervision)

Александр Дьяконов

29 мая 2020 года

План

Самообучение

Основные термины:

pretext task, downstream task

Разные постановки предварительных задач

Современные тенденции в самообучении:

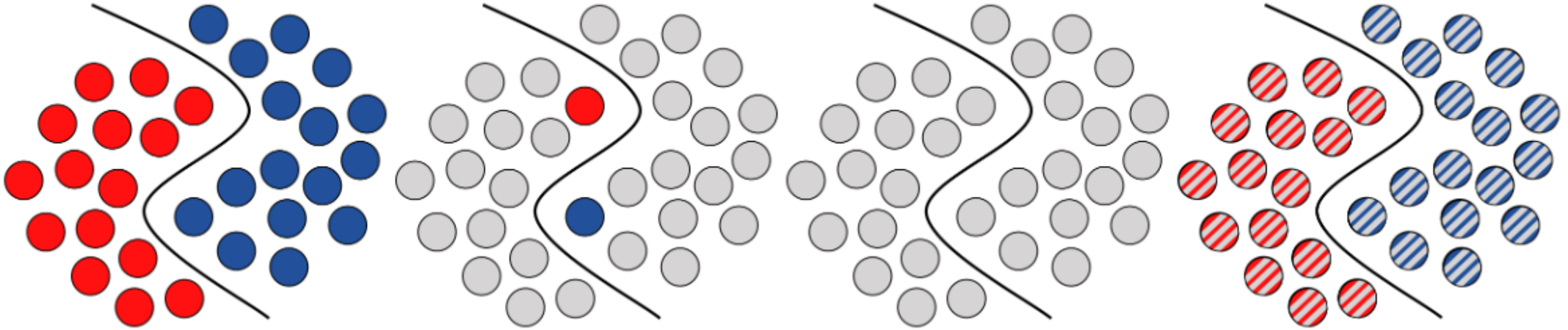
информационные функции ошибки,

сравнительное обучение

Self-Supervised Learning

– создать свою задачу с метками на имеющихся (неразмеченных) данных

обычно часть данных объявляют метками и предсказывают по оставшейся части



(a) Supervised (b) Semi-Supervised (c) Unsupervised (d) Self-Supervised

Figure 2: Illustrations of the four presented deep learning strategies - The red and dark blue circles represent labeled data points of different classes. The light grey circles represent unlabeled data points. The black lines define the underlying decision boundary between the classes. The striped circles represent datapoints which ignore and use the label information at different stages of the training process.

https://arxiv.org/pdf/2002.08721.pdf

Self-Supervised Learning

pretext task

не наша задача, но позволяет получить хорошее представление

метки (pseudo labels) в ней получаются «автоматически»

здесь обычно генерация признаков – representation

downstream task

решение задачи с помощью сгенерированных признаков простой моделью

в изображениях чаще классификация, детектирование, сегментация

здесь разметка, как правило, ручная (human-annotated labels)

дальше схема эксперимента:

обучаем признаковое представление, потом донастраиваем на них логистическую регрессию (или 1NN)

Цель – обучить представление (признаковое) без ручной разметки

Self-Supervised Learning

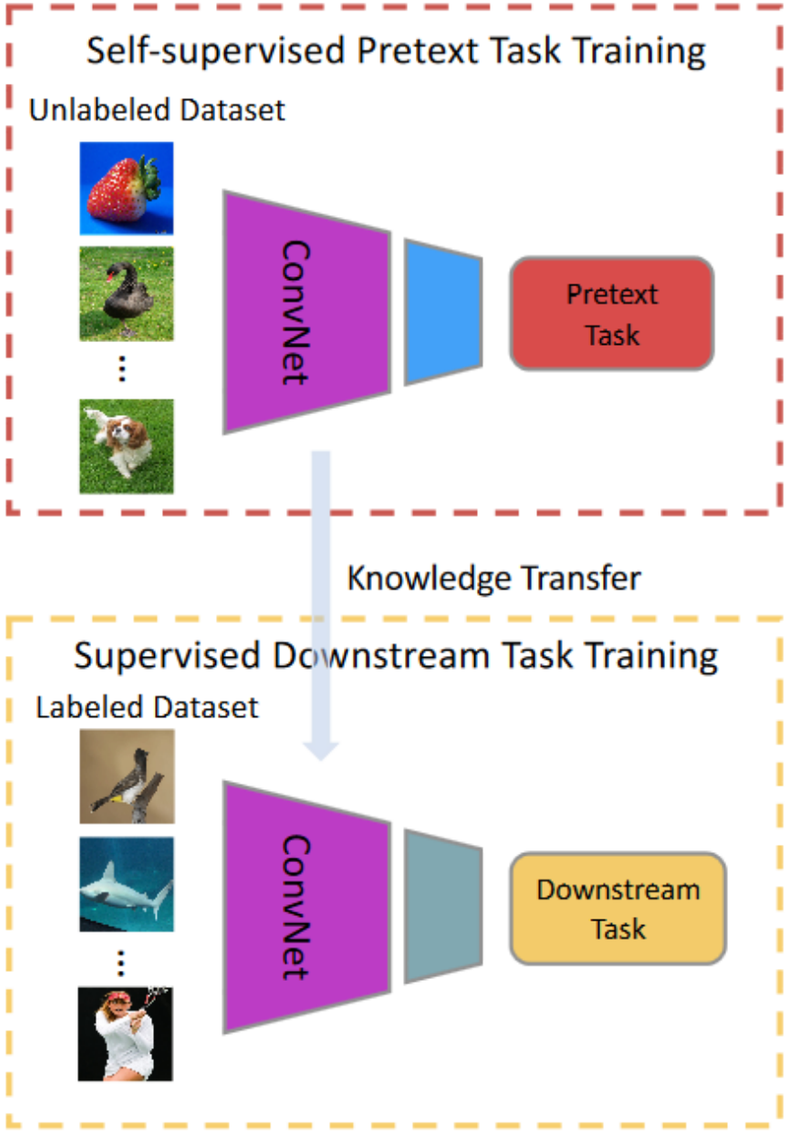


Fig. 1. The general pipeline of self-supervised learning. The visual feature is learned through the process of training ConvNets to solve a pre-defined pretext task. After self-supervised pretext task training finished, the learned parameters serve as a pre-trained model and are transferred to other downstream computer vision tasks by fine-tuning. The performance on these downstream tasks is used to evaluate the quality of the learned features. During the knowledge transfer for downstream tasks, the general features from only the first several layers are unusually transferred to downstream tasks.

https://arxiv.org/pdf/1902.06162.pdf

Self-Supervised Learning

Причины:

- **меток может не быть**
- **разметка может быть дорогой**
- **есть много неразмеченных данных**
- **соответствует реальному обучению (человека)**

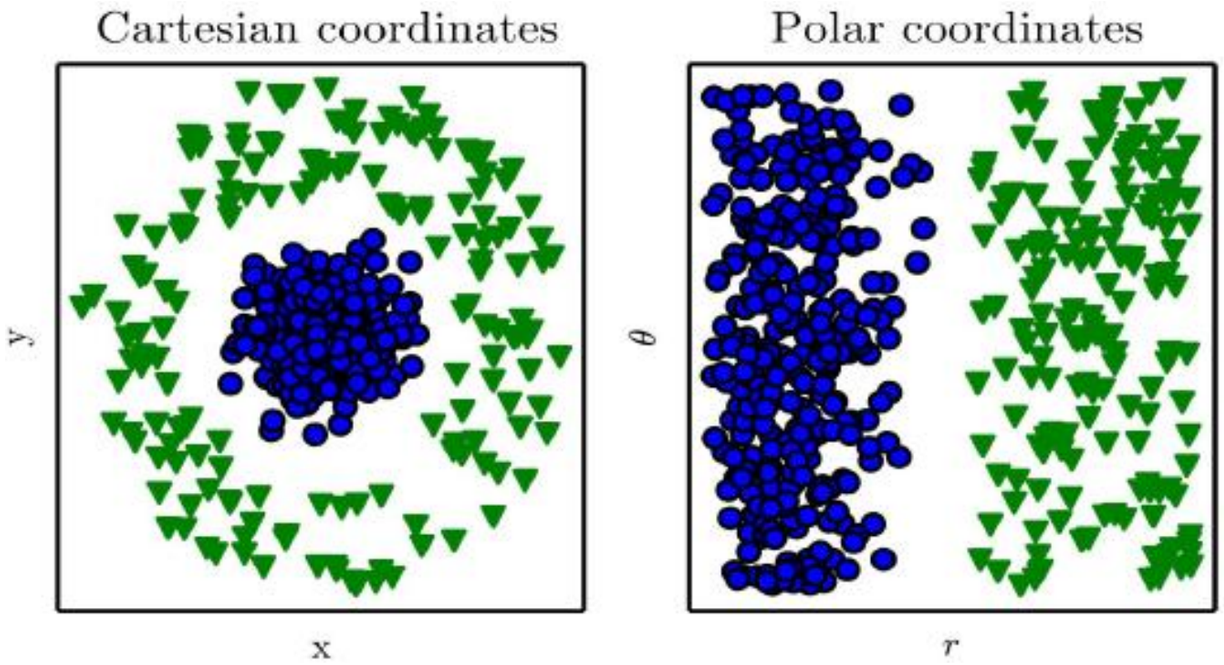
Мотивы:

надежда, что эти признаки окажутся полезными для решения других задач (downstream tasks)

получение представления данных (representations)

предобучение (pre-training)

Что такое представление (representations)



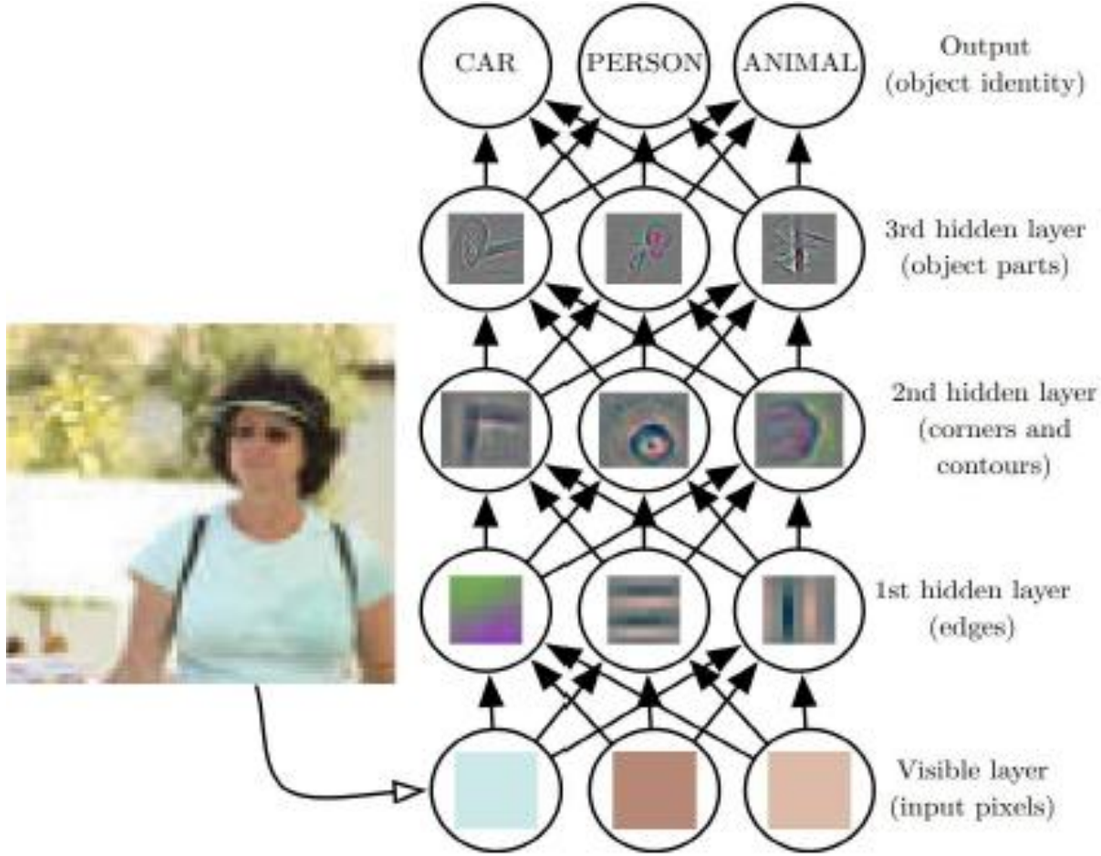


рис. Goodfellow

Transfer Learning

in Language	in Vision
BERT	CPCv2
word2vec	MoCo
fastText	MoCo v2
...	...
	в этих слайдах

Задачи (pretext learning tasks)

~ понимание изображение «на высоком уровне»

обучение средних слоёв нейронных сетей

автокодировщики / генеративные модели	
Предсказывание контекста (predicting context)	C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. InInter-national Conference on Computer Vision (ICCV), 2015 https://arxiv.org/pdf/1505.05192.pdf
Детекция поворота (predicting image rotation)	S. Gidaris, P. Singh, and N. Komodakis. Unsupervised rep-resentation learning by predicting image rotations. InIn-ternational Conference on Learning Representations (ICLR),2018. https://arxiv.org/pdf/1803.07728.pdf
Образец (Exemplar)	A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, andT. Brox. Discriminative unsupervised feature learning withconvolutional neural networks. InAdvances in Neural Infor-mation Processing Systems (NIPS), 2014. https://arxiv.org/pdf/1406.6909.pdf
Головоломка (Jigsaw)	M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. InEuropean Con-ference on Computer Vision (ECCV), 2016. https://arxiv.org/pdf/1603.09246.pdf

Кластеризация (DeepCluster)	M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. European Conference on Computer Vision (ECCV), 2018. https://arxiv.org/abs/1807.05520
Контекстные кодировщики (Context Encoder) / image inpainting	D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. InConference on Computer Vision and Pattern Recognition(CVPR), 2016. https://arxiv.org/abs/1604.07379
Раскраска изображений (image colorization)	R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. InEuropean Conference on Computer Vision (ECCV), 2016, https://richzhang.github.io/colorization/
Расщеплённые автокодировщики (Split-brain autoencoders)	R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. InConference on Computer Vision and Pattern Recognition(CVPR), 2017. https://arxiv.org/abs/1611.09842
Сегментация, порождённая движением (motion segmentation prediction)	D. Pathak, R. B. Girshick, P. Doll ar, T. Darrell, and B. Hariharan. Learning features by watching objects move. In Conference on Computer Vision and Pattern Recognition(CVPR), 2017 https://arxiv.org/abs/1612.06370
Окружающие звуки (ambient sound)	A. Owens, J. Wu, J. H. McDermott, W. T. Freeman, and A. Torralba. Ambient sound provides supervision for visuallearning.ECCV, 2016 https://arxiv.org/abs/1608.07017

Подсчёт примитивов (counting visual primitives)	M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. InInternational Conferenceon Computer Vision (ICCV), 2017. https://arxiv.org/abs/1708.06734
CPC	A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding https://arxiv.org/abs/1807.03748

Predicting (spatial) context

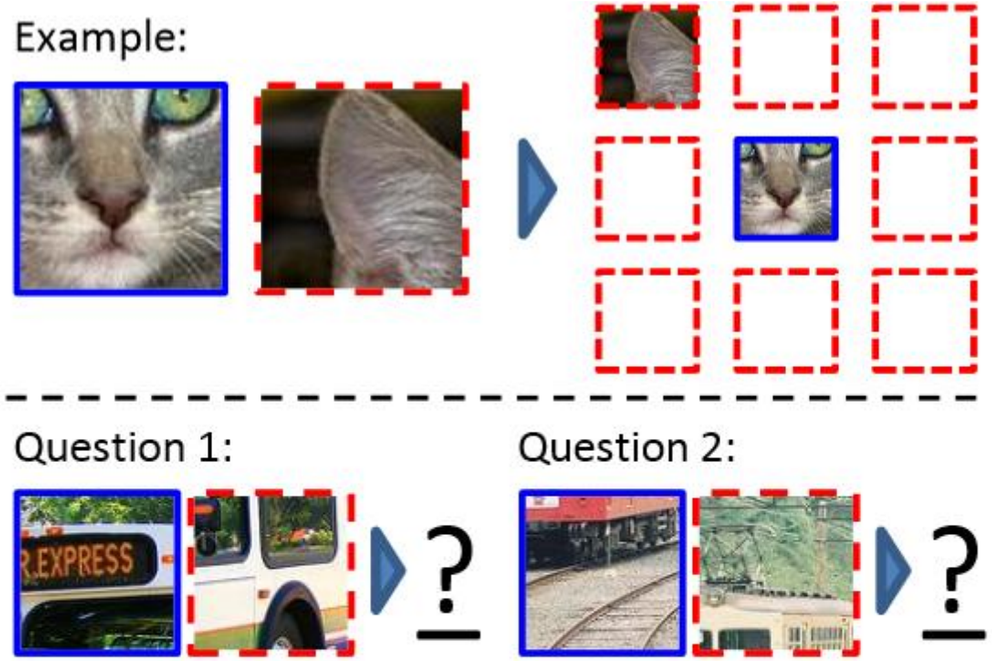


Figure 1. Our task for learning patch representations involves randomly sampling a patch (blue) and then one of eight possible neighbors (red). Can you guess the spatial configuration for the two pairs of patches? Note that the task is much easier once you have recognized the object!

Answer key: Q1: Bottom right Q2: Top center

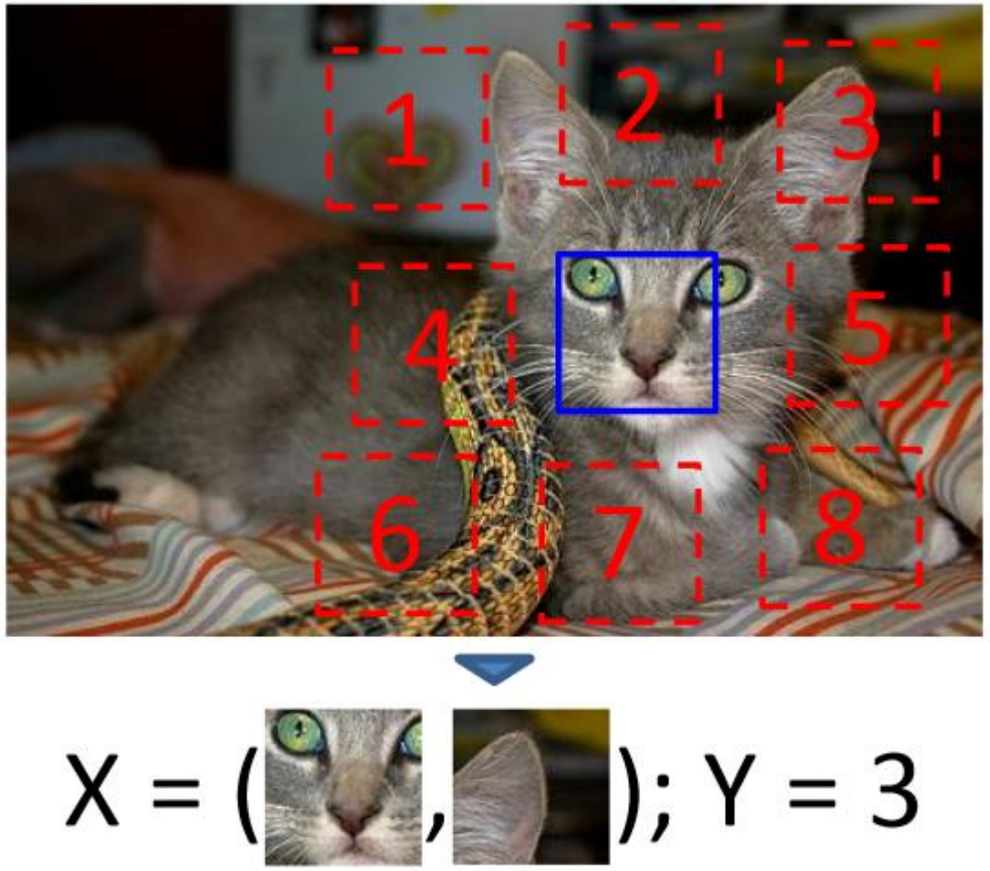


Figure 2. The algorithm receives two patches in one of these eight possible spatial arrangements, without any context, and must then classify which configuration was sampled.

C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. InInternational Conference on Computer Vision (ICCV), 2015 https://arxiv.org/pdf/1505.05192.pdf

Predicting (spatial) context

Идея из текстовой модели «skip-gram»

первый патч выбирается случайно, второй – около него

между патчами – зазоры, патчи случайно смещены

два канала забиваются шумом

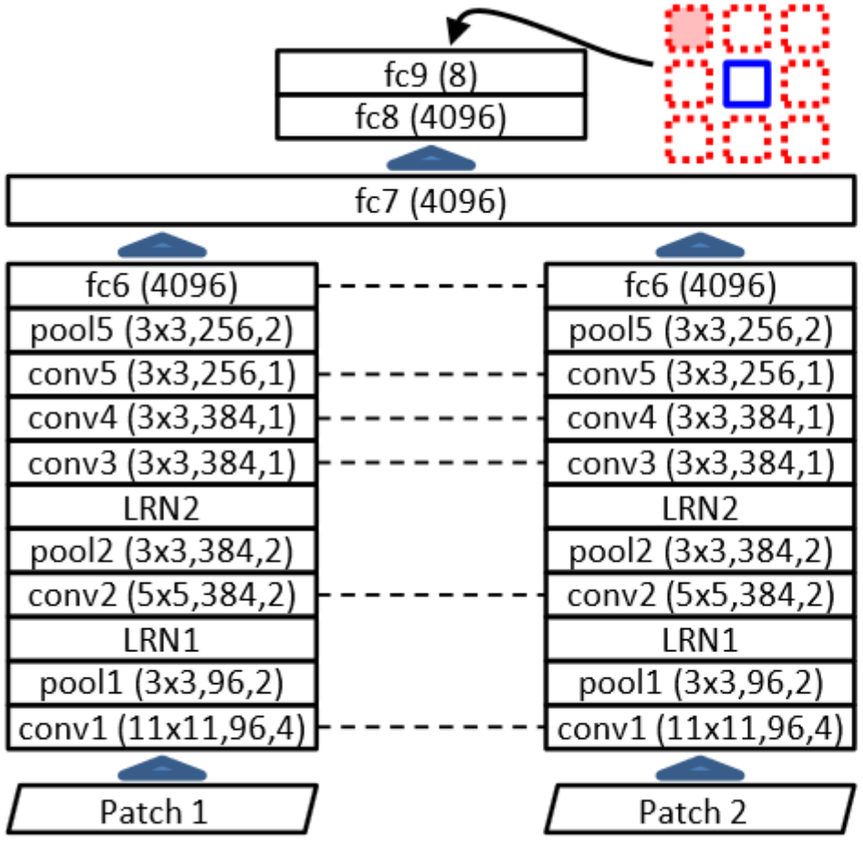


Figure 3. Our architecture for pair classification. Dotted lines indicate shared weights. 'conv' stands for a convolution layer, 'fc' stands for a fully-connected one, 'pool' is a max-pooling layer, and 'LRN' is a local response normalization layer. Numbers in parentheses are kernel size, number of outputs, and stride (fc layers have only a number of outputs). The LRN parameters follow [32]. All conv and fc layers are followed by ReLU nonlinearities, except fc9 which feeds into a softmax classifier.

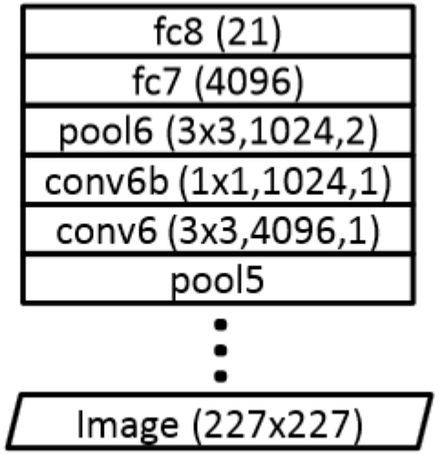


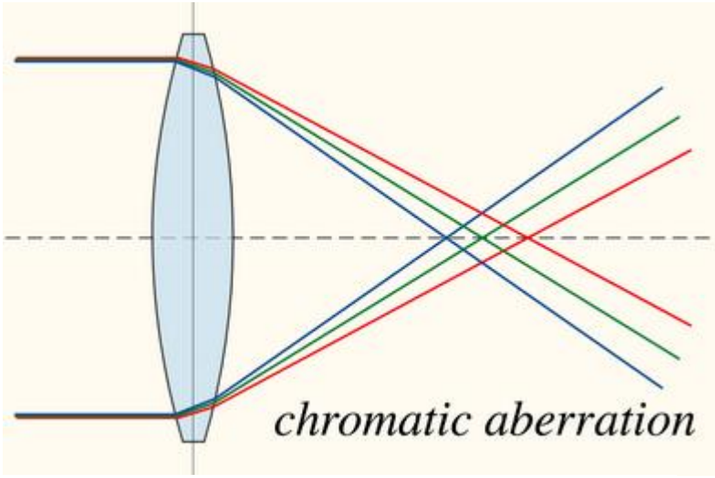
Figure 6. Our architecture for Pascal VOC detection. Layers from conv1 through pool5 are copied from our patch-based network (Figure 3). The new 'conv6' layer is created by converting the fc6 layer into a convolution layer. Kernel sizes, output units, and stride are given in parentheses, as in Figure 3.

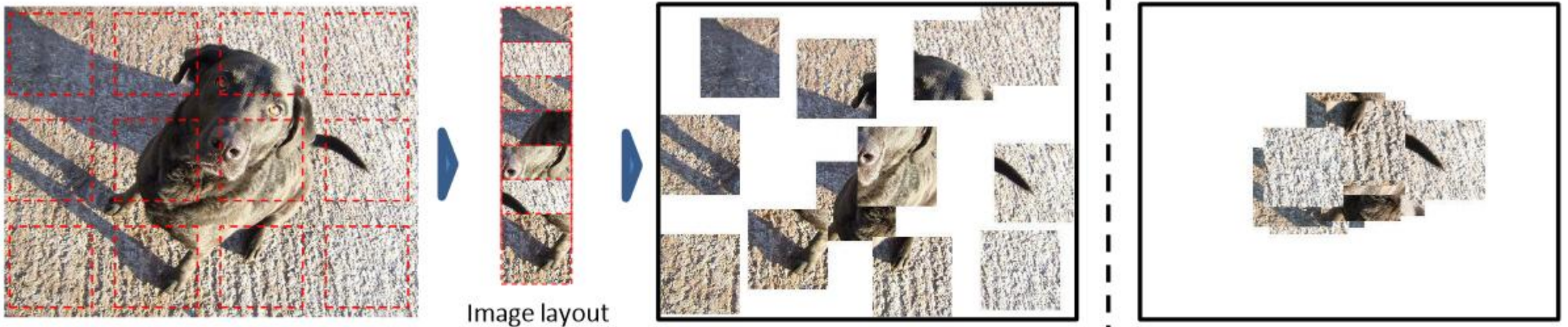
AlexNet-style architectures

Predicting (spatial) context

Зачем нужно что-то делать с цветом…
по нему можно определить позицию патча

это явление – хроматическая аберрация





Initial layout, with sampled patches in red | Image layout is discarded | We can recover image layout automatically | Cannot recover layout with color removed

Figure 5. We trained a network to predict the absolute (x, y) coordinates of randomly sampled patches. Far left: input image. Center left: extracted patches. Center right: the location the trained network predicts for each patch shown on the left. Far right: the same result after our color projection scheme. Note that the far right patches are shown *after* color projection; the operation's effect is almost unnoticeable.

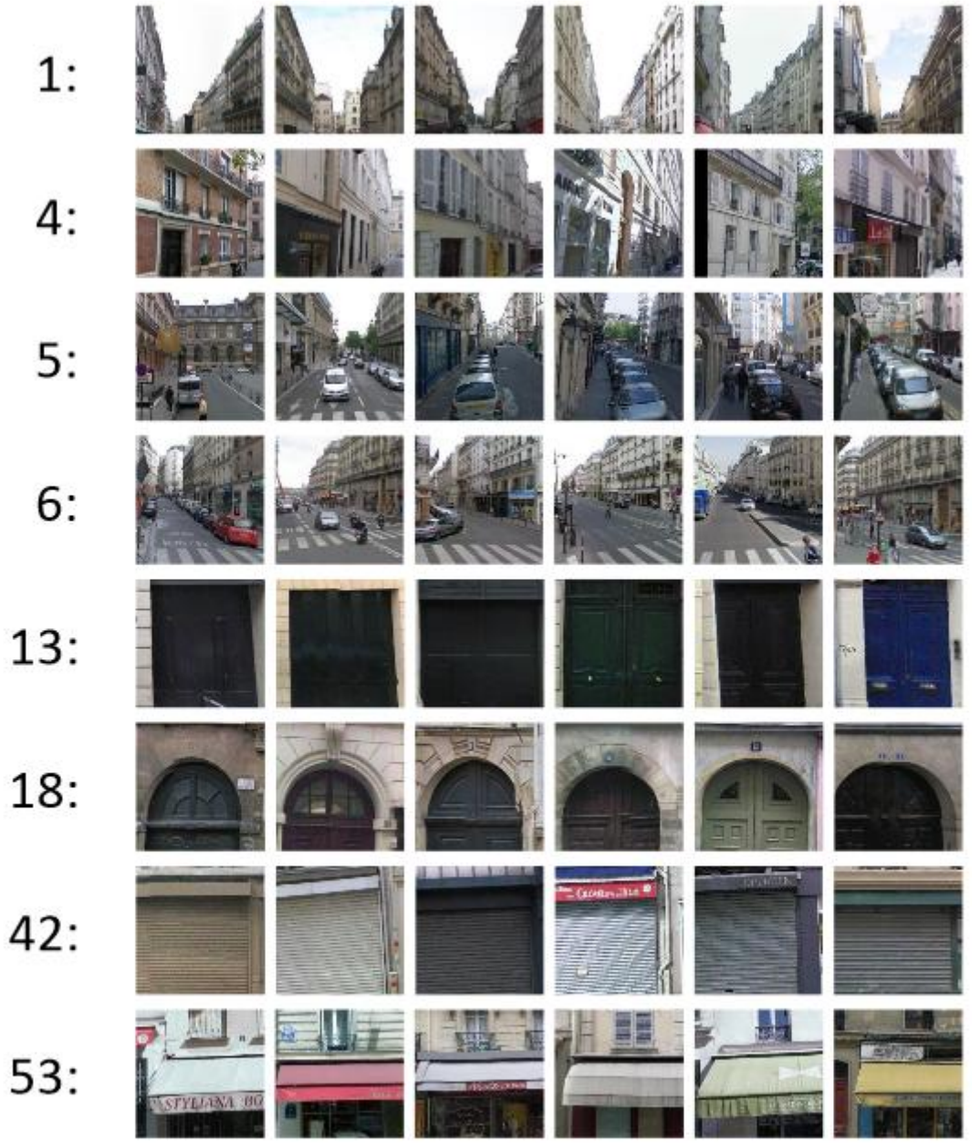


Figure 8. Clusters discovered and automatically ranked via our algorithm (§ 4.5) from the Paris Street View dataset.

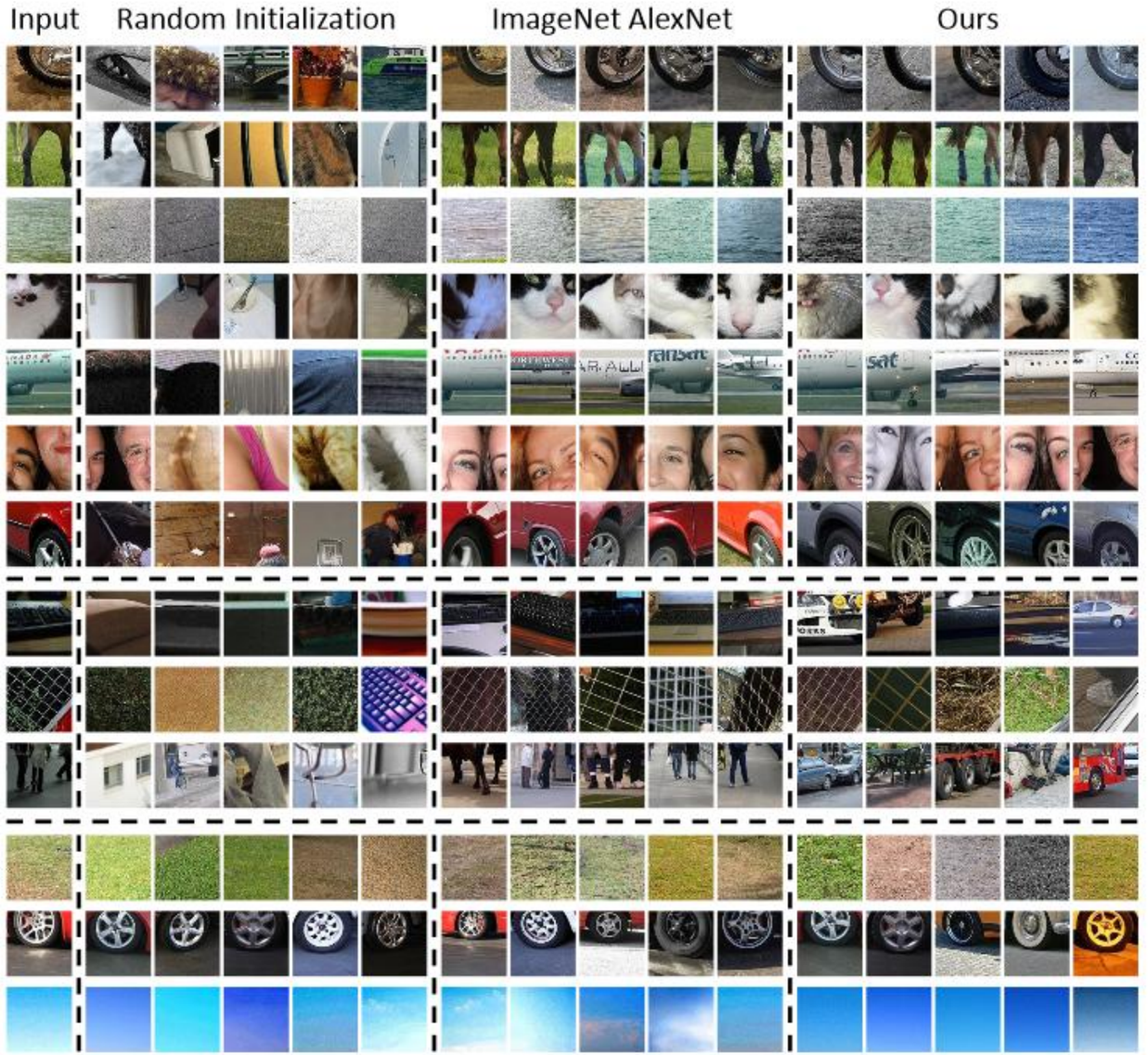
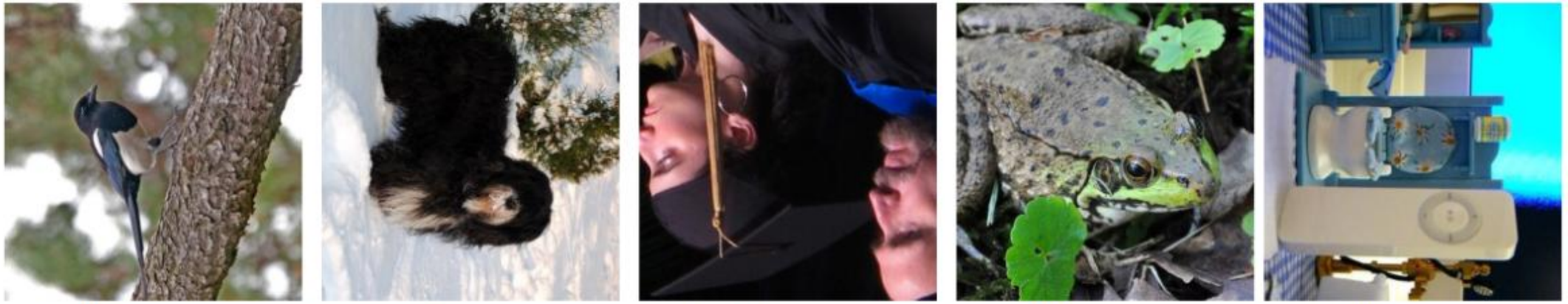


Figure 4. Examples of patch clusters obtained by nearest neighbors. The query patch is shown on the far left. Matches are for three different features: fc6 features from a random initialization of our architecture, AlexNet fc7 after training on labeled ImageNet, and the fc6 features learned from our method. Queries were chosen from 1000 randomly-sampled patches. The top group is examples where our algorithm performs well; for the middle AlexNet outperforms our approach; and for the bottom all three features work well.

Predicting image rotation



90° rotation 270° rotation 180° rotation 0° rotation 270° rotation

Figure 1: Images rotated by random multiples of 90 degrees (e.g., 0, 90, 180, or 270 degrees). The core intuition of our self-supervised feature learning approach is that if someone is not aware of the concepts of the objects depicted in the images, he cannot recognize the rotation that was applied to them.

S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In International Conference on Learning Representations (ICLR),2018.
https://arxiv.org/pdf/1803.07728.pdf

Predicting image rotation

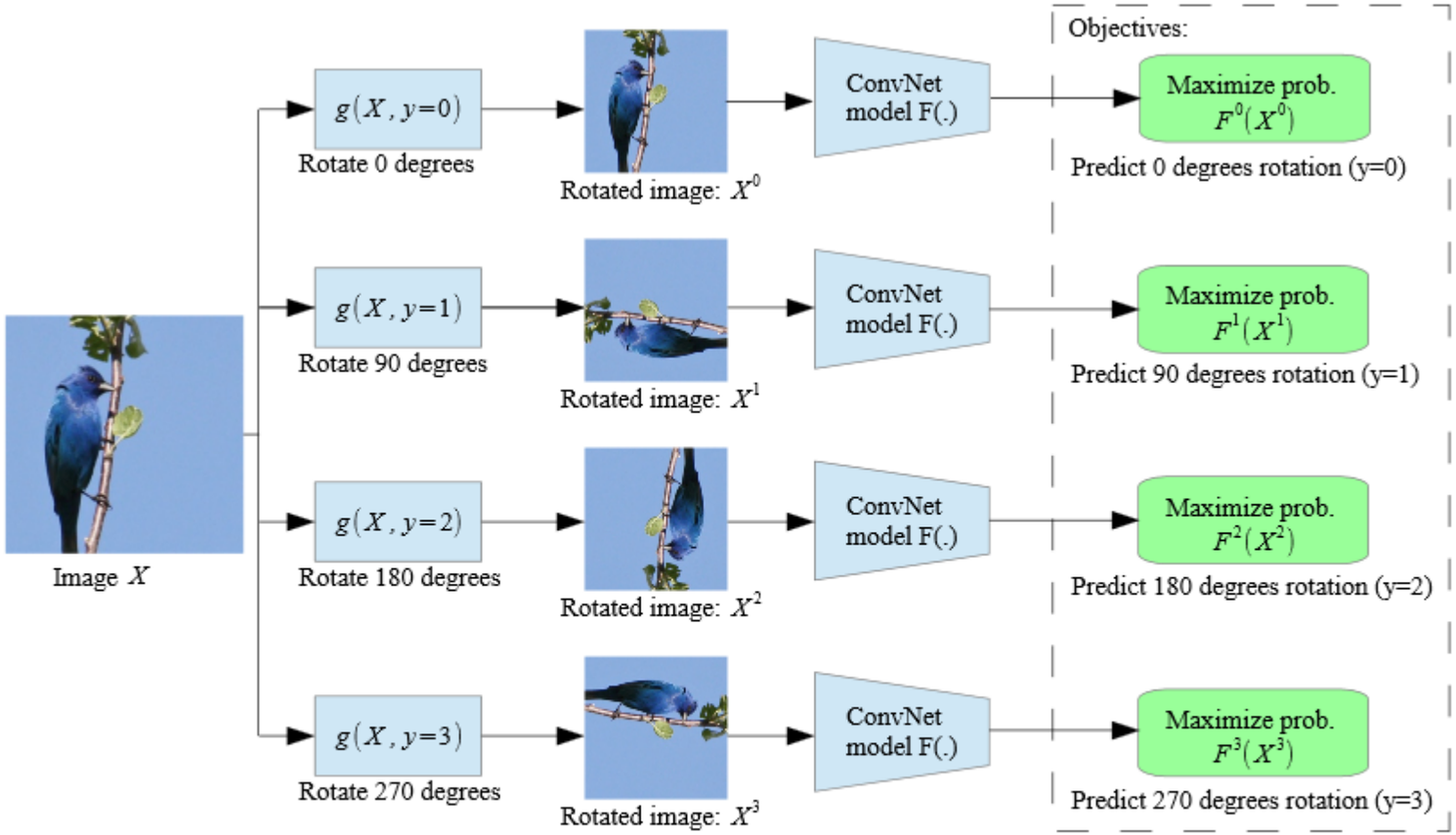
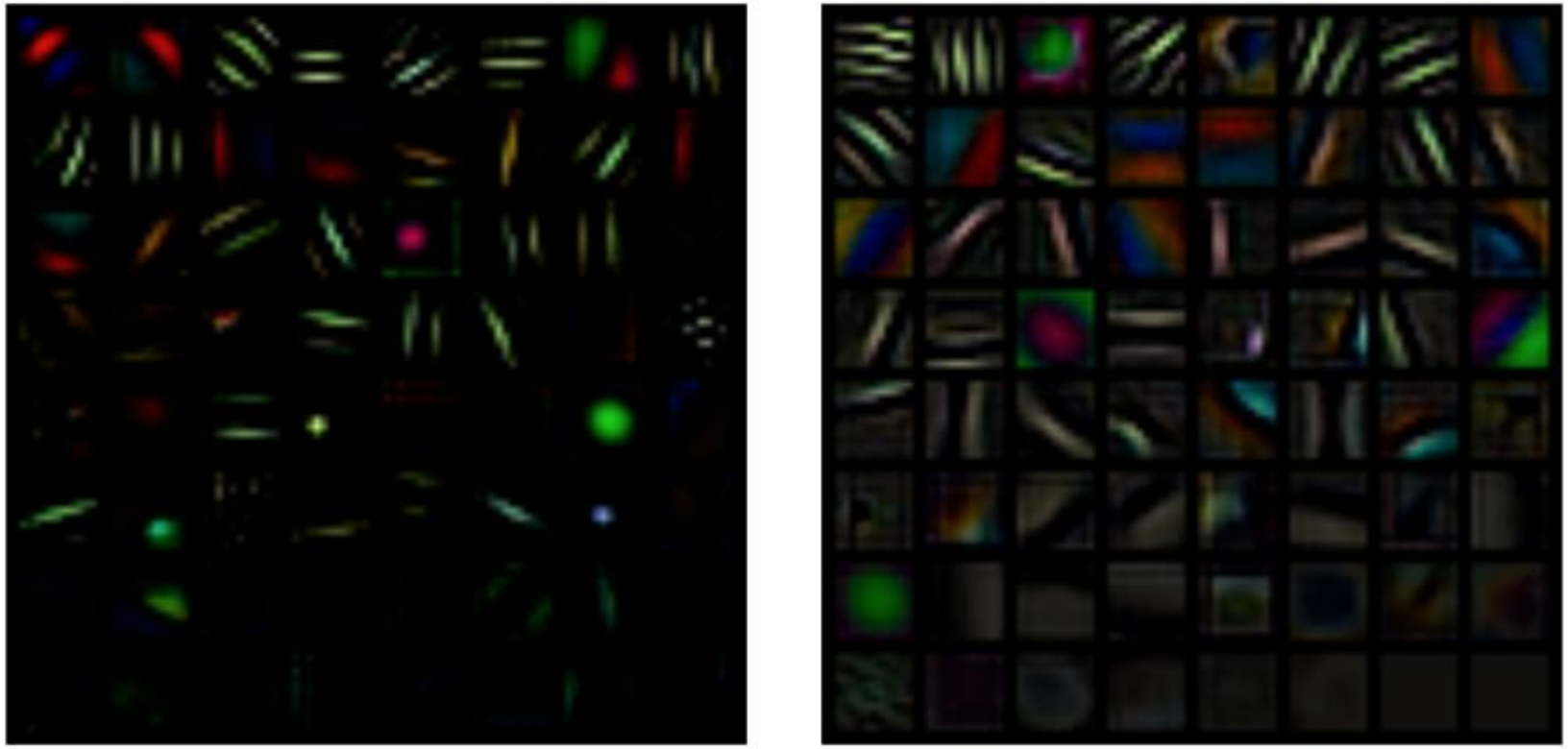


Figure 2: Illustration of the self-supervised task that we propose for semantic feature learning. Given four possible geometric transformations, the 0, 90, 180, and 270 degrees rotations, we train a ConvNet model $F(.)$ to recognize the rotation that is applied to the image that it gets as input. $F^y(X^{y^*})$ is the probability of rotation transformation y predicted by model $F(.)$ when it gets as input an image that has been transformed by the rotation transformation y^*.

Predicting image rotation



(a) **Supervised** (b) **Self-supervised to recognize rotations**

Figure 4: First layer filters learned by a AlexNet model trained on **(a)** the supervised object recognition task and **(b)** the self-supervised task of recognizing rotated images. We observe that the filters learned by the self-supervised task are mostly oriented edge filters on various frequencies and, remarkably, they seem to have more variety than those learned on the supervised task.

Predicting image rotation

Table 2: Exploring the quality of the self-supervised learned features w.r.t. the number of recognized rotations. For all the entries we trained a non-linear classifier with 3 fully connected layers (similar to Table 1) on top of the feature maps generated by the 2nd conv. block of a RotNet model with 4 conv. blocks in total. The reported results are from CIFAR-10.

# Rotations	Rotations	CIFAR-10 Classification Accuracy
4	0°, 90°, 180°, 270°	**89.06**
8	0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°	88.51
2	0°, 180°	87.46
2	90°, 270°	85.52

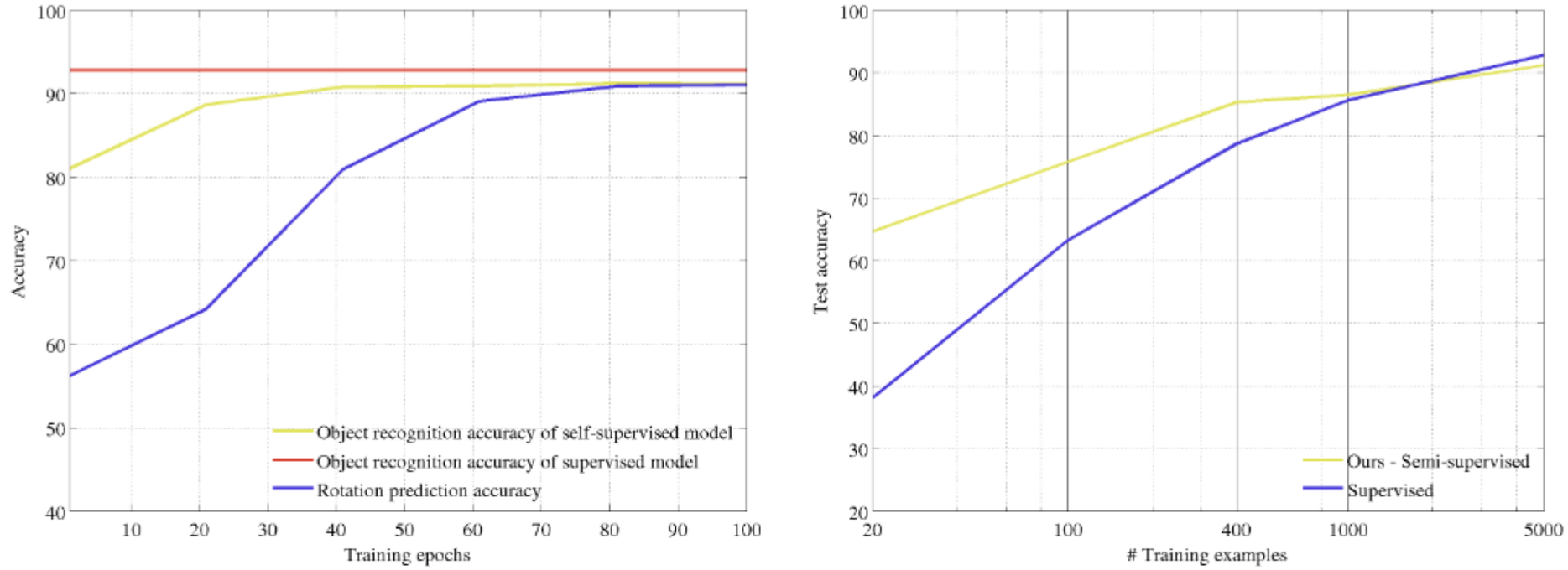
не везде повороты имеют смысл, пример – биологические изображения клеток

инварианты относительно поворотов

Input images on the models

(a) **Attention maps of supervised model** (b) **Attention maps of our self-supervised model**

Figure 3: Attention maps generated by an AlexNet model trained (**a**) to recognize objects (supervised), and (**b**) to recognize image rotations (self-supervised). In order to generate the attention map of a conv. layer we first compute the feature maps of this layer, then we raise each feature activation on the power p, and finally we sum the activations at each location of the feature map. For the conv. layers 1, 2, and 3 we used the powers $p = 1$, $p = 2$, and $p = 4$ respectively. For visualization of our self-supervised model's attention maps for all the rotated versions of the images see Figure 6 in appendix A.



(a) (b)

Figure 5: **(a)** Plot with the rotation prediction accuracy and object recognition accuracy as a function of the training epochs used for solving the rotation prediction task. The red curve is the object recognition accuracy of a fully supervised model (a NIN model), which is independent from the training epochs on the rotation prediction task. The yellow curve is the object recognition accuracy of an object classifier trained on top of feature maps learned by a *RotNet* model at different snapshots of the training procedure. **(b)** Accuracy as a function of the number of training examples per category in CIFAR-10. *Ours semi-supervised* is a NIN model that the first 2 conv. blocks are *RotNet* model that was trained in a self-supervised way on the entire training set of CIFAR-10 and the 3rd conv. block along with a prediction linear layer that was trained with the object recognition task only on the available set of labeled images.

Table 5: **Task Generalization: ImageNet top-1 classification with linear layers**. We compare our unsupervised feature learning approach with other unsupervised approaches by training logistic regression classifiers on top of the feature maps of each layer to perform the 1000-way ImageNet classification task, as proposed by Zhang et al. (2016a). All weights are frozen and feature maps are spatially resized (with adaptive max pooling) so as to have around 9000 elements. All approaches use AlexNet variants and were pre-trained on ImageNet without labels except the ImageNet labels and Random entries.

Method	Conv1	Conv2	Conv3	Conv4	Conv5
ImageNet labels	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Random rescaled Krähenbühl et al. (2015)	17.5	23.0	24.5	23.2	20.6
Context (Doersch et al., 2015)	16.2	23.3	30.2	31.7	29.6
Context Encoders (Pathak et al., 2016b)	14.1	20.7	21.0	19.8	15.5
Colorization (Zhang et al., 2016a)	12.5	24.5	30.4	31.5	30.3
Jigsaw Puzzles (Noroozi & Favaro, 2016)	18.2	28.8	34.0	33.9	27.1
BIGAN (Donahue et al., 2016)	17.7	24.5	31.0	29.9	28.0
Split-Brain (Zhang et al., 2016b)	17.7	29.3	35.4	35.2	32.8
Counting (Noroozi et al., 2017)	18.0	30.6	34.3	32.5	25.7
(Ours) RotNet	**18.8**	**31.7**	**38.7**	**38.2**	**36.5**

Predicting image rotation: дальнейшее использование идеи

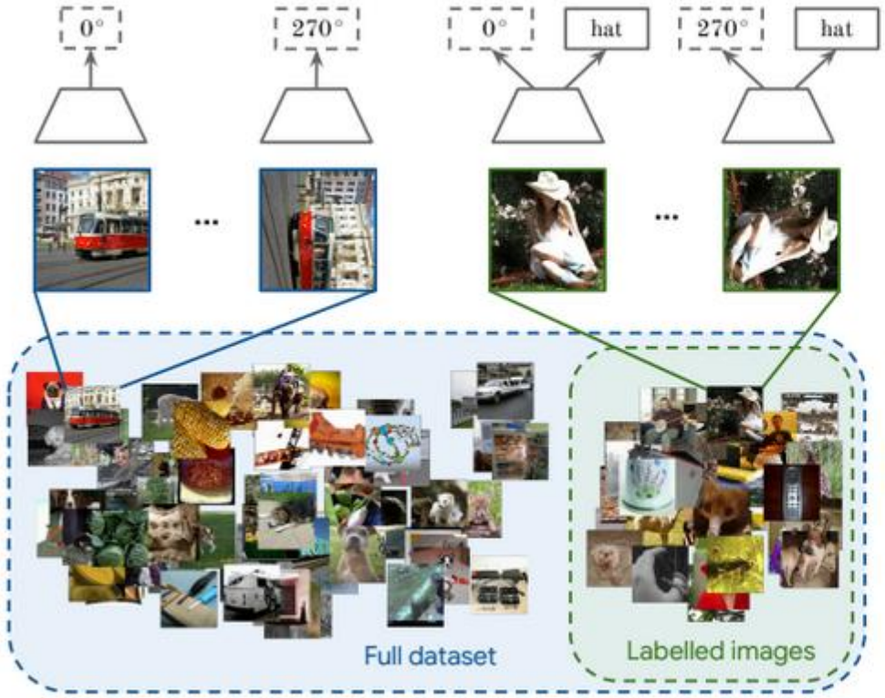


Figure 1. A schematic illustration of one of the proposed self-supervised semi-supervised techniques: S^4L-Rotation. Our model makes use of both labeled and unlabled images. The first step is to create four input images for any image by rotating it by 0°, 90°, 180° and 270° (inspired by (Gidaris et al., 2018)). Then, we train a network that predicts which rotation was applied to all these images and predicts semantic labels of annotated images.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, Lucas Beyer «S4L: Self-Supervised Semi-Supervised Learning»
https://drive.google.com/file/d/0B4M2IUVyJzS4eEdQZmlIT1BEeUVDbVFDM3dHM0JicDBtT3VZ/view

Predicting image rotation: дальнейшее использование идеи

The methods we consider have a learning objective of the following form:

$$\min_{\theta} \; \mathcal{L}_l(D_l, \theta) + w\mathcal{L}_u(D_u, \theta), \qquad (1)$$

where $\mathcal{L}_l$ the a standard cross-entropy classification loss, w is a non-negative scalar weight and θ are the parameters of the model. In the proposed S^4L framework, we propose using any self-supervision technique as unsupervised loss $\mathcal{L}_u$. This objective can be extended to multiple losses $\mathcal{L}_u$.

Exemplar

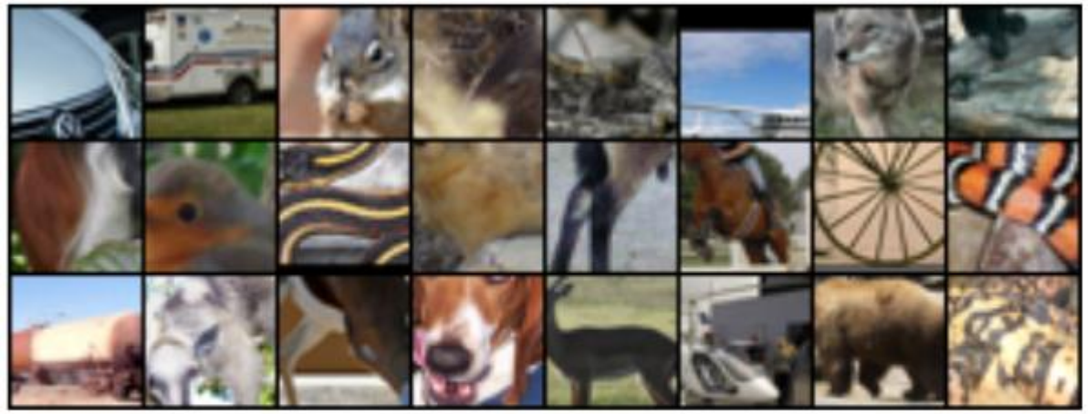


Fig. 1. Exemplary patches sampled from the STL unlabeled dataset which are later augmented by various transformations to obtain surrogate data for the CNN training.

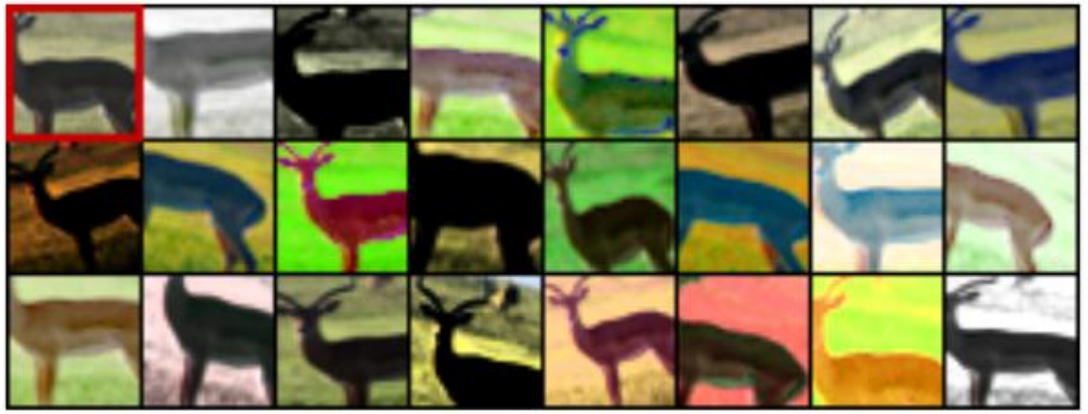
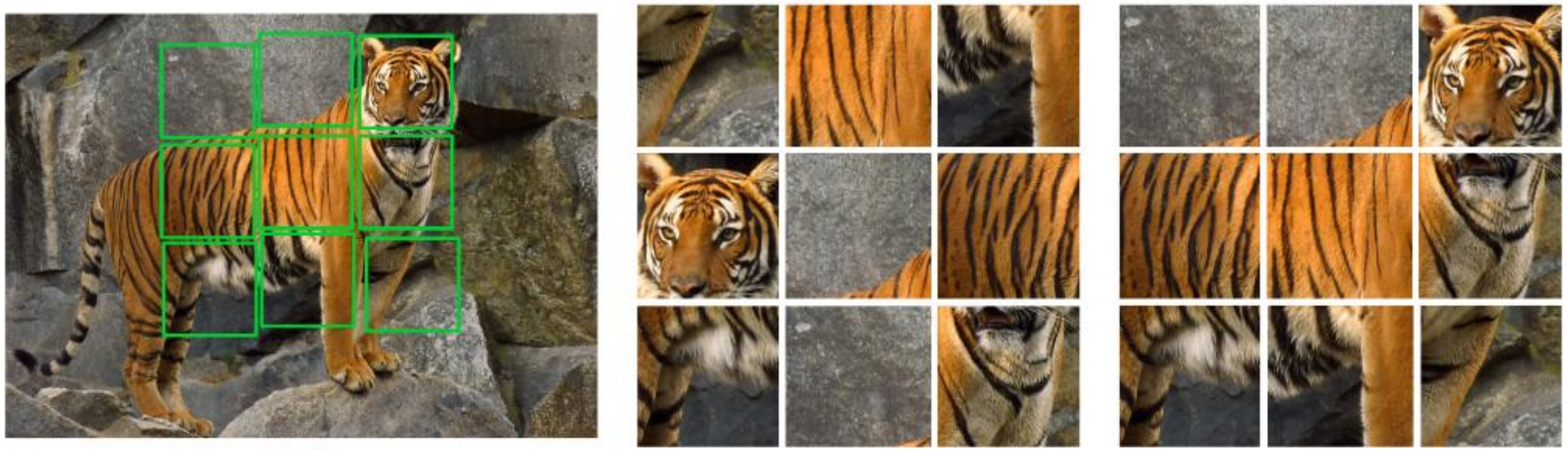


Fig. 2. Several random transformations applied to one of the patches extracted from the STL unlabeled dataset. The original ('seed') patch is in the top left corner.

Выбрать «интересные участки», применить аугментации (translation, scaling, rotation, contrast, color), решить задачу классификации, класс – номер референсного изображения.

A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox Discriminative unsupervised feature learning with convolutional neural networks. In Advances in Neural Information Processing Systems (NIPS), 2014.https://arxiv.org/pdf/1406.6909.pdf

Головоломка (Jigsaw)

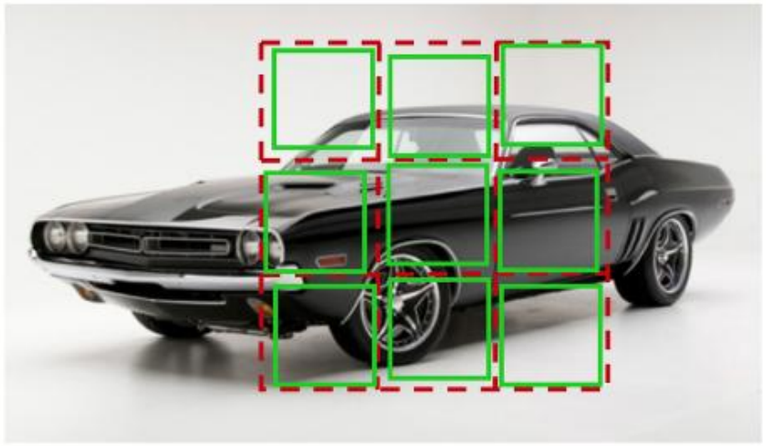


(a) (b) (c)

Fig. 1: Learning image representations by solving Jigsaw puzzles. (a) The image from which the tiles (marked with green lines) are extracted. (b) A puzzle obtained by shuffling the tiles. Some tiles might be directly identifiable as object parts, but others are ambiguous (*e.g.*, have similar patterns) and their identification is much more reliable when all tiles are jointly evaluated. In contrast, with reference to (c), determining the relative position between the central tile and the top two tiles from the left can be very challenging [10].

M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In European Conference on Computer Vision (ECCV), 2016. https://arxiv.org/pdf/1603.09246.pdf

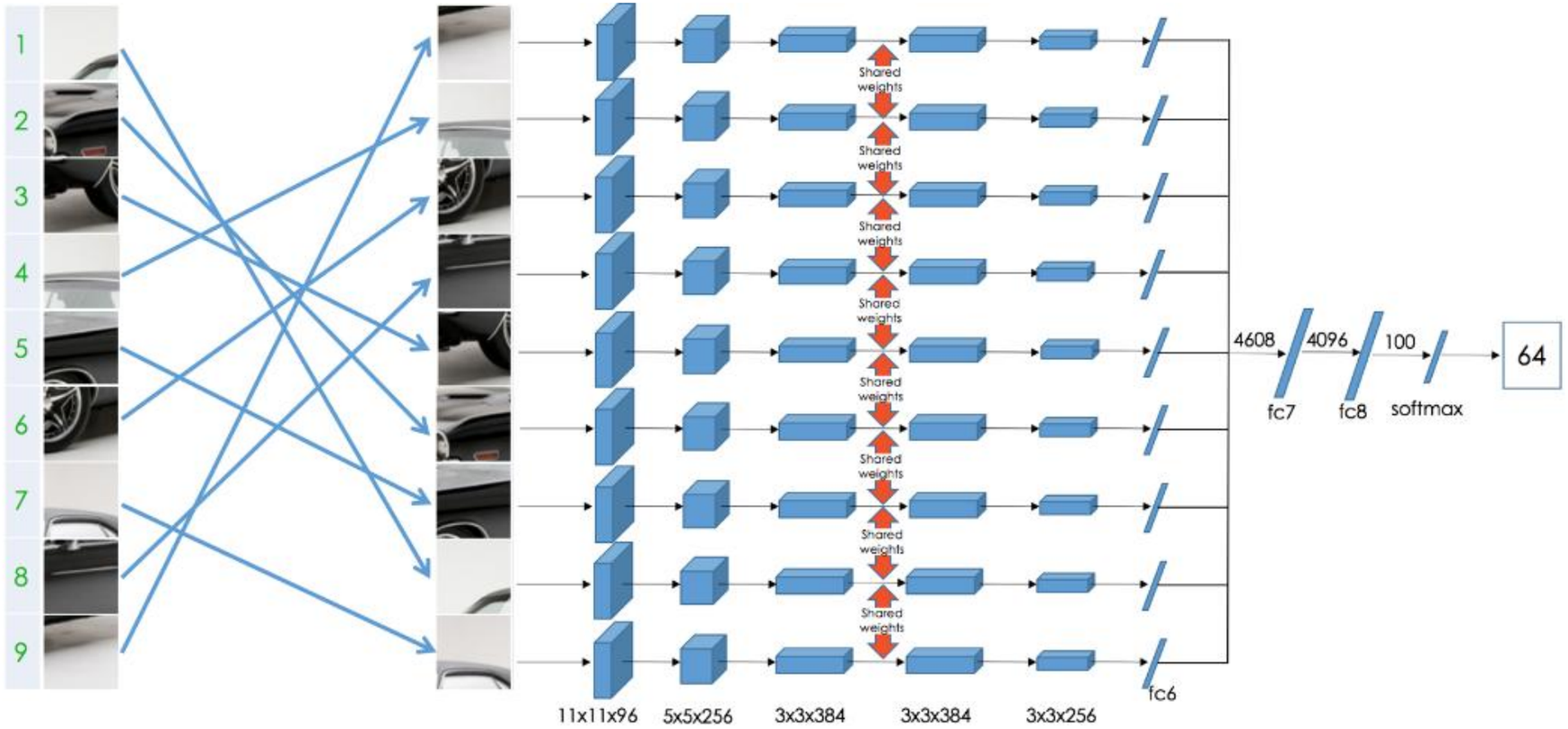
Головоломка (Jigsaw)



Permutation Set

index	permutation
64	9,4,6,8,3,2,5,1,7

Reorder patches according to the selected permutation



окно 225×225 из изображения делим на 9 частей, действуем одной из заданных перестановок, задача – определить номер перестановки

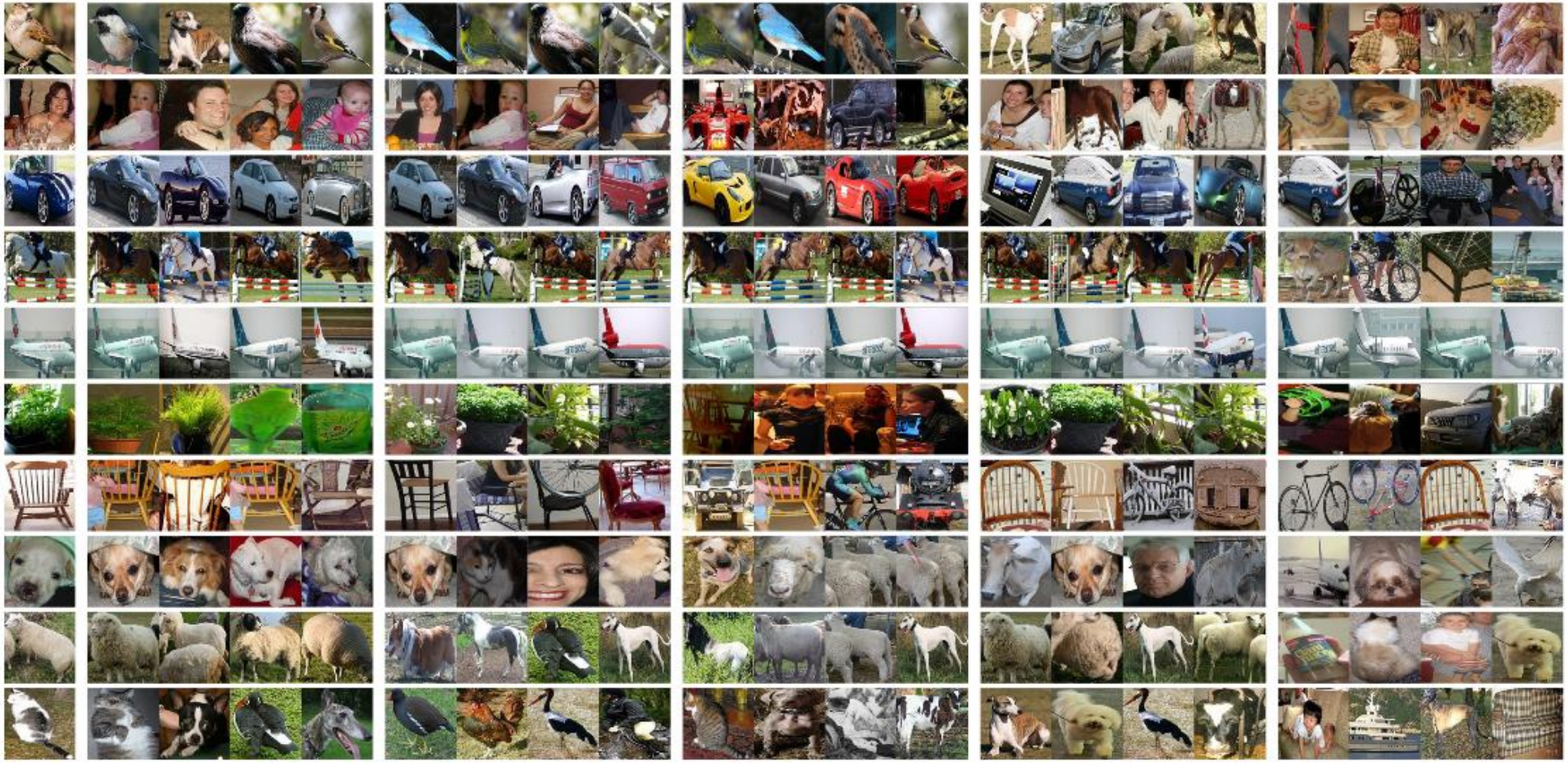
не показаны пулинг и нелинейность

Головоломка (Jigsaw)

независимая нормализация патчей

зазоры между патчами

Chromatic Aberration – разные способы устранения



(a) (b) (c) (d) (e) (f)

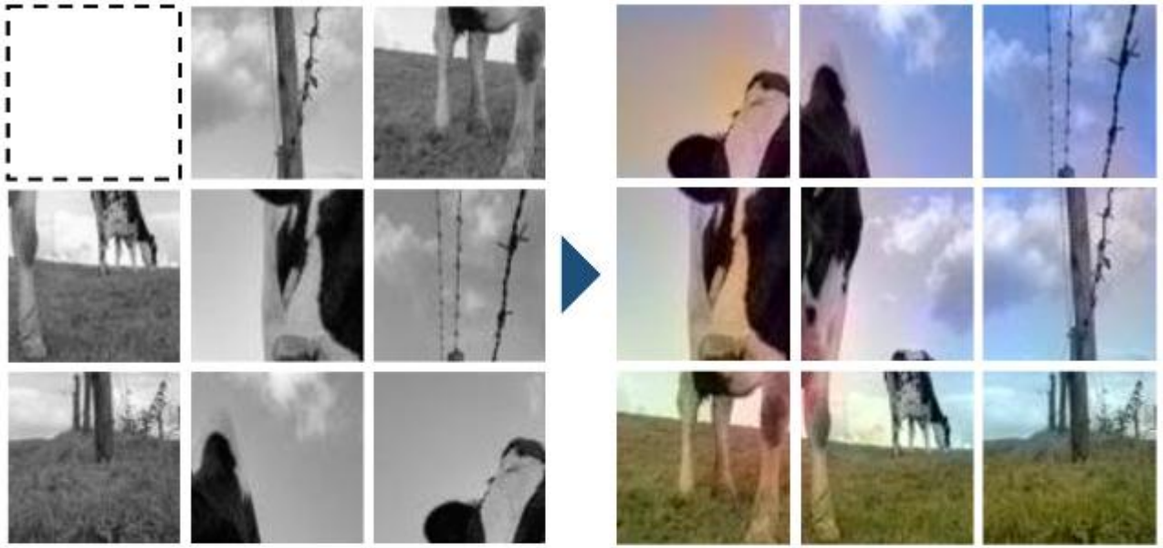
Fig. 5: Image retrieval (qualitative evaluation). (a) query images; (b) top-4 matches with AlexNet; (c) top-4 matches with the CFN trained without blocking chromatic aberration; (d) top-4 matches with Doersch *et al.* [10]; (e) top-4 matches with Wang and Gupta [39]; (f) top-4 matches with AlexNet with random weights.

Головоломка (Jigsaw): дальнейшее развитие

(a) main image (b) different image (c) Jigsaw (d) Jigsaw++

Figure 8: Illustrations of the pretext task Jigsaw and Jigsaw++ - The Jigsaw pretext task consists of solving a simple Jigsaw puzzle generated from the main image. Jigsaw++ augments the Jigsaw puzzle by adding in parts of a different image. The illustrations are inspired by [36].

Головоломка (Jigsaw): дальнейшее развитие



(a) (b)

Figure 1. **Learning image representations by completing damaged jigsaw puzzles.** We sample 3-by-3 patches from an image and create damaged jigsaw puzzles. (a) is the puzzles after shuffling the patches, removing one patch, and decolorizing. We push a network to recover the original arrangement, the missing patch, and the color of the puzzles. (b) shows the outputs; while the pixel-level predictions are in *ab* channels, we visualize with their original L channels for the benefit of the reader.

Kim, D., Cho, D., Yoo, D., and Kweon, I. S. Learning Image Representations by Completing Damaged Jigsaw Puzzles. In WACV 2018, 2018. https://arxiv.org/abs/1802.01880

Головоломка (Jigsaw): дальнейшее развитие

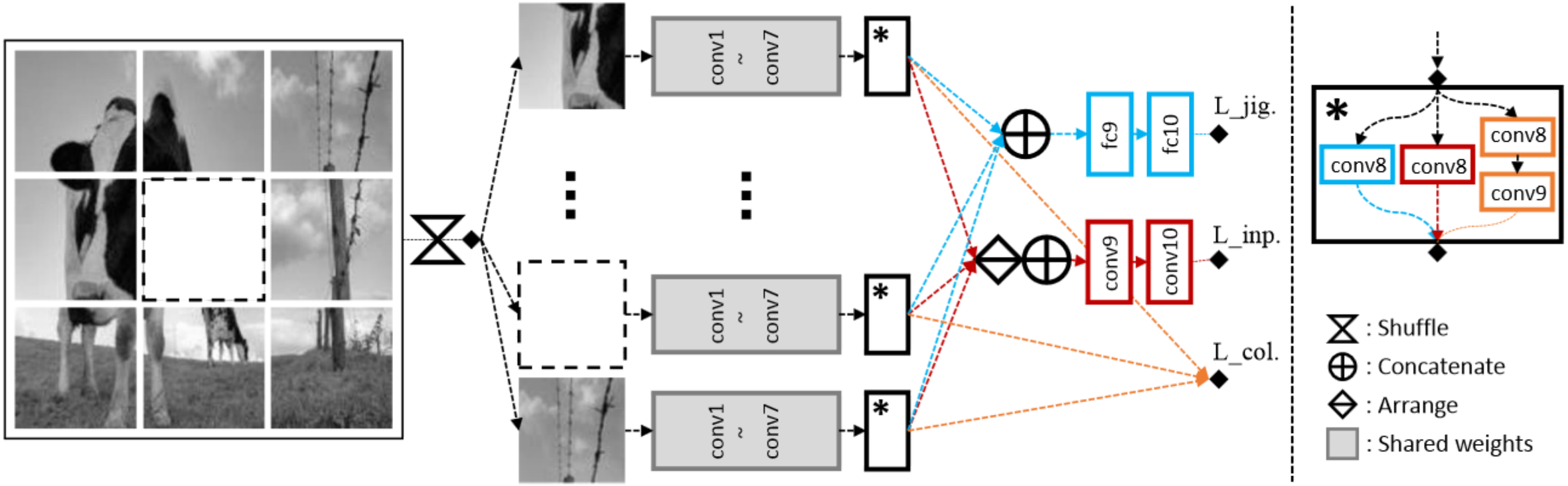


Figure 3. **The architecture for "Completing damaged jigsaw puzzles".** It is a 9-tower siamese network. The shared tower(colored in gray) consists of AlexNet *conv*1-7 layers. note *fc*6-7 are converted into equivalent *conv*6-7 layers for the pixel-level outputs. The task branches for jigsaw puzzle, inpainting, and colorization are marked in blue, red, and orange, respectively. The learned shared tower is used for transfer learning on downstream tasks.

Головоломка (Jigsaw): дальнейшее развитие

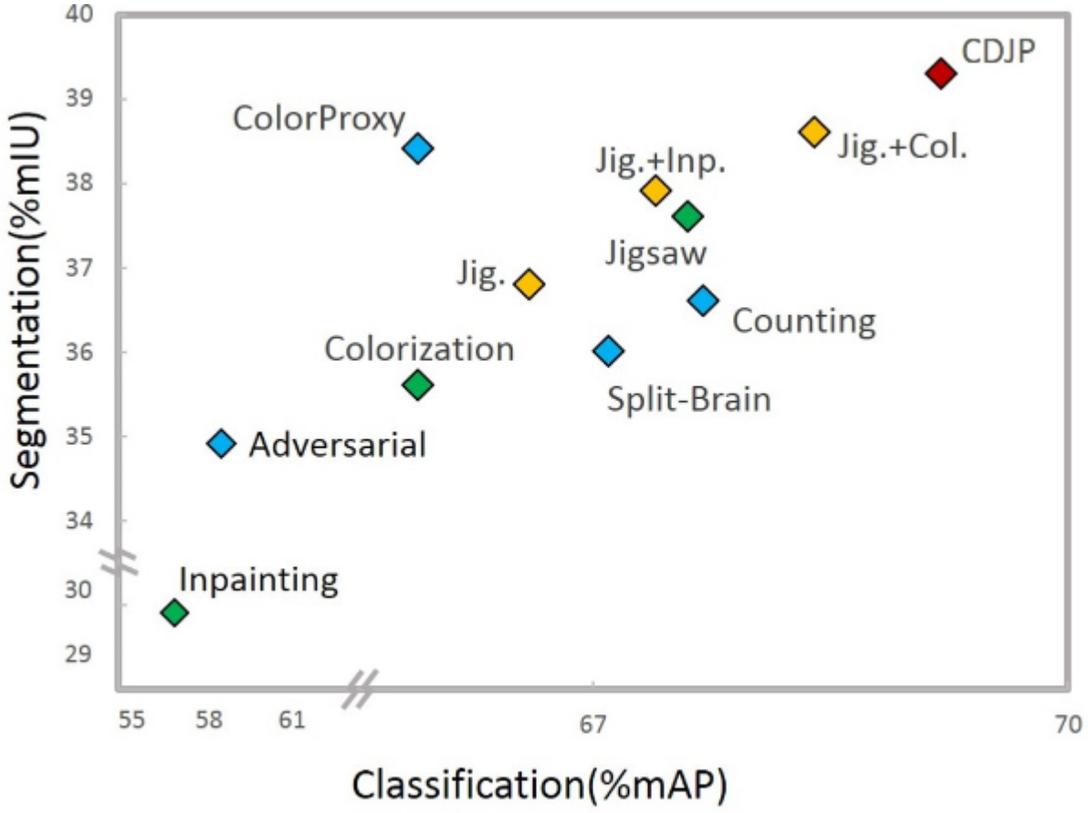


Figure 4. **Summarization of performances of different self-supervised learning methods and combinations.** We compare the state-of-the-art methods(Table. 2), our final method (CDJP), and each involved tasks in our final method and the simple combination(Table. 4). The involved tasks, their original versions, the simple combination, the other existing methods, and our final method are marked in orange, green, gray, blue and red, respectively. Note that $Jig.$ is what we reproduced in our architecture.

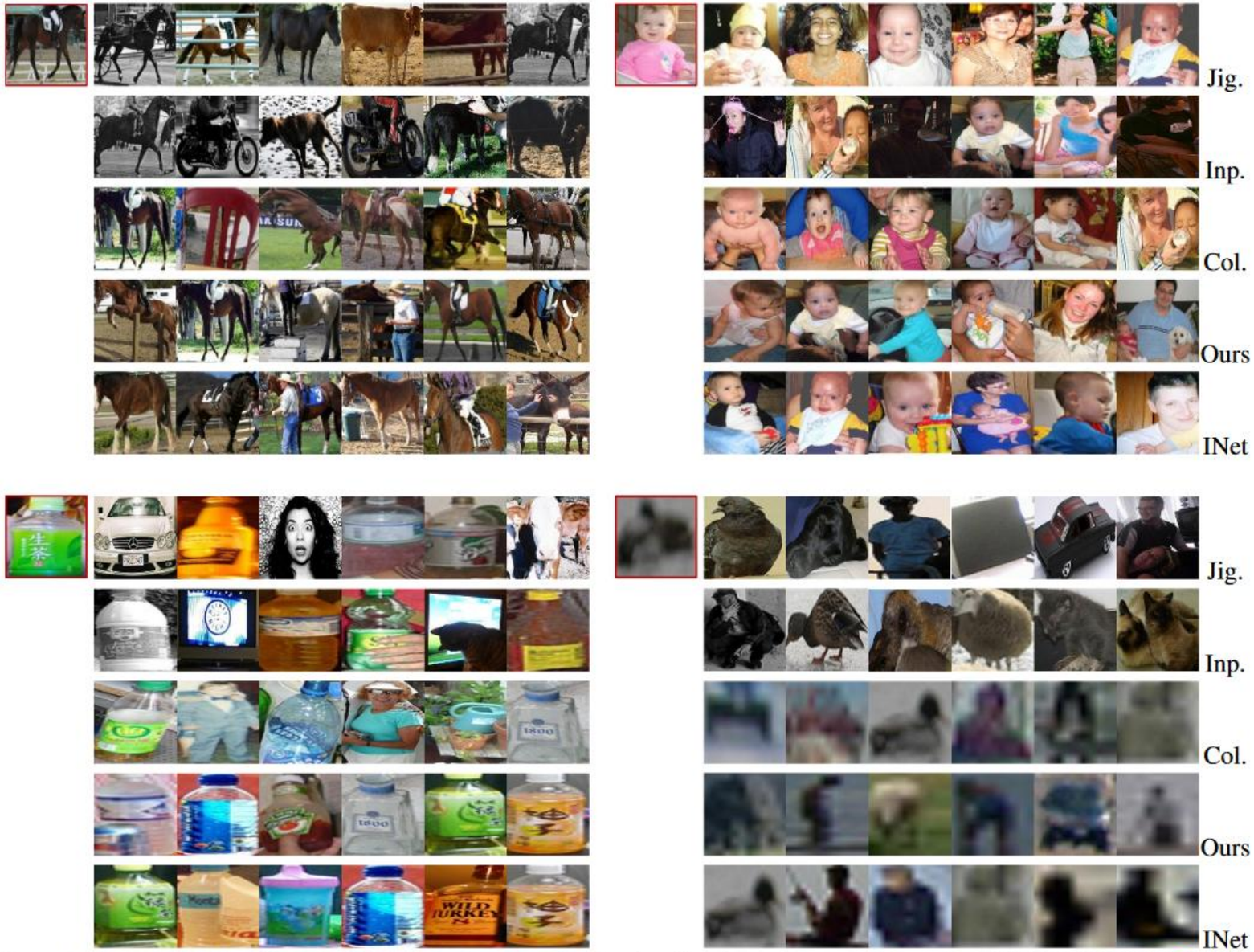


Figure 5. **Nearest Neighbor Search.** We perform image retrieval on the object instances cropped from the PASCAL VOC 2012 [8] *trainval* dataset. The query images are in red boxes. Down from the top rows are the retrieval results of jigsaw puzzle, inpainting, colorization, our method, and ImageNet classification, respectively.

Кластеризация (DeepCluster)

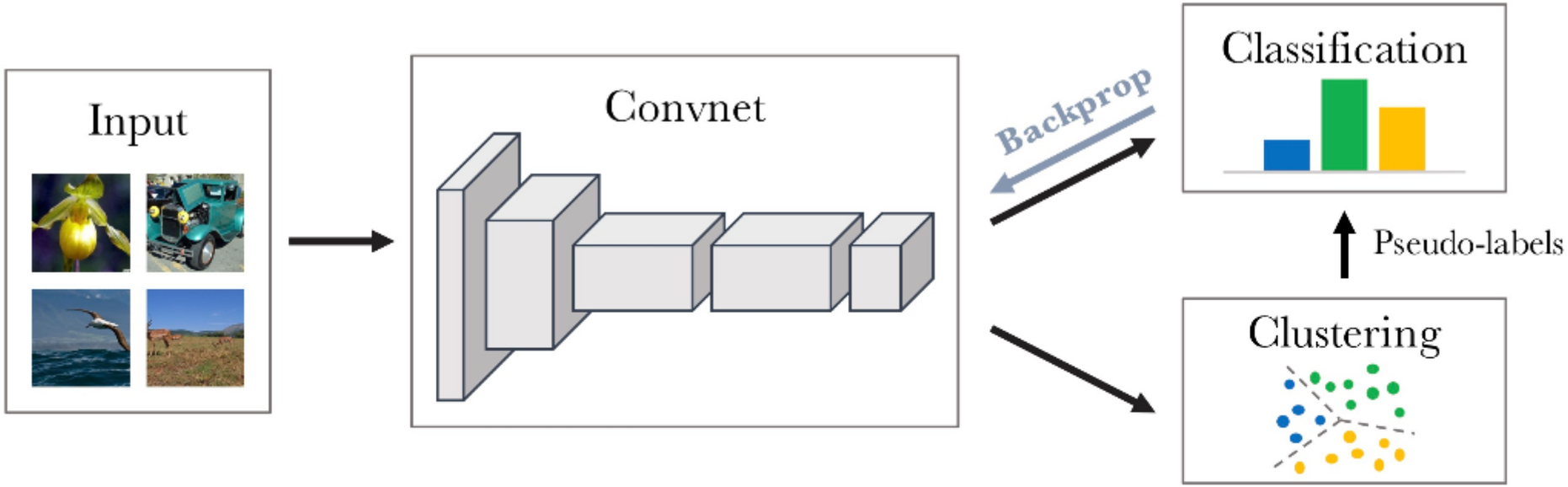


Fig. 1: Illustration of the proposed method: we iteratively cluster deep features and use the cluster assignments as pseudo-labels to learn the parameters of the convnet.

M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. European Conference on Computer Vision (ECCV), 2018.
https://arxiv.org/abs/1807.05520

Кластеризация (DeepCluster)

интересный факт: если случайно задать веса AlexNet и настраивать только последний слой, то качество на ImageNet 12% (случайное – 0.1%)

кластеризуем выходы какого-нибудь слоя (k-means)
используем номера кластеров как метки
делаем это итеративно

избегать пустых кластеров
сэмплирование при котором будет баланс представителей классов
Sobel filters – для удаления информации о цвете и увеличения контраста
аугментация данных

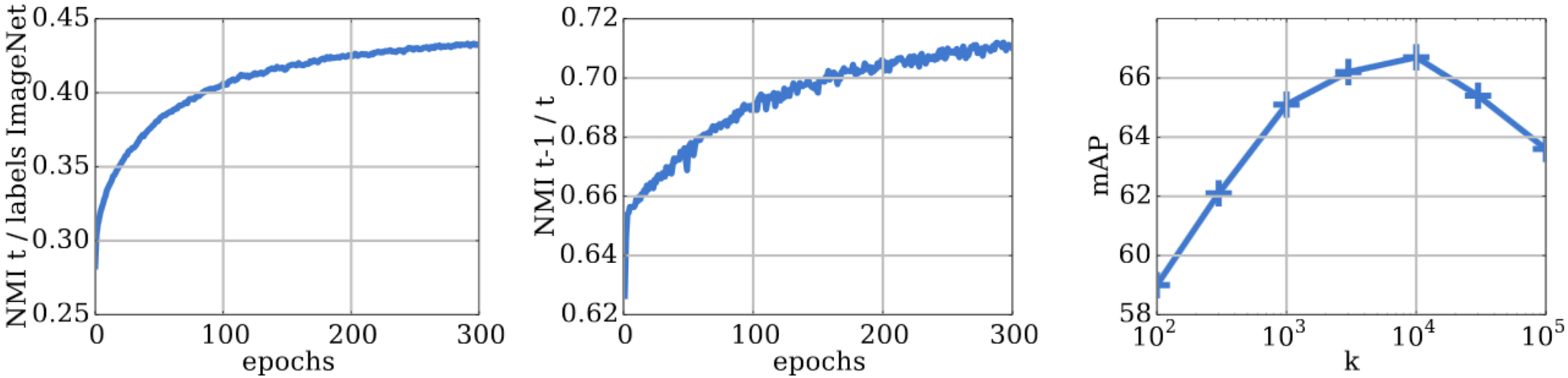
Перекластеризация (overclustering) – см. ниже – число кластеров должно быть больше числа классов.

Кластеризация (DeepCluster)

Normalized Mutual Information (NMI) –

$$\mathrm{NMI}(A;B) = \frac{\mathrm{I}(A;B)}{\sqrt{\mathrm{H}(A)\mathrm{H}(B)}}$$

I – mutual information, H – entropy



(a) Clustering quality (b) Cluster reassignment (c) Influence of k

Fig. 2: Preliminary studies. (a): evolution of the clustering quality along training epochs; (b): evolution of cluster reassignments at each clustering step; (c): validation mAP classification performance for various choices of k.

Кластеризация (DeepCluster)

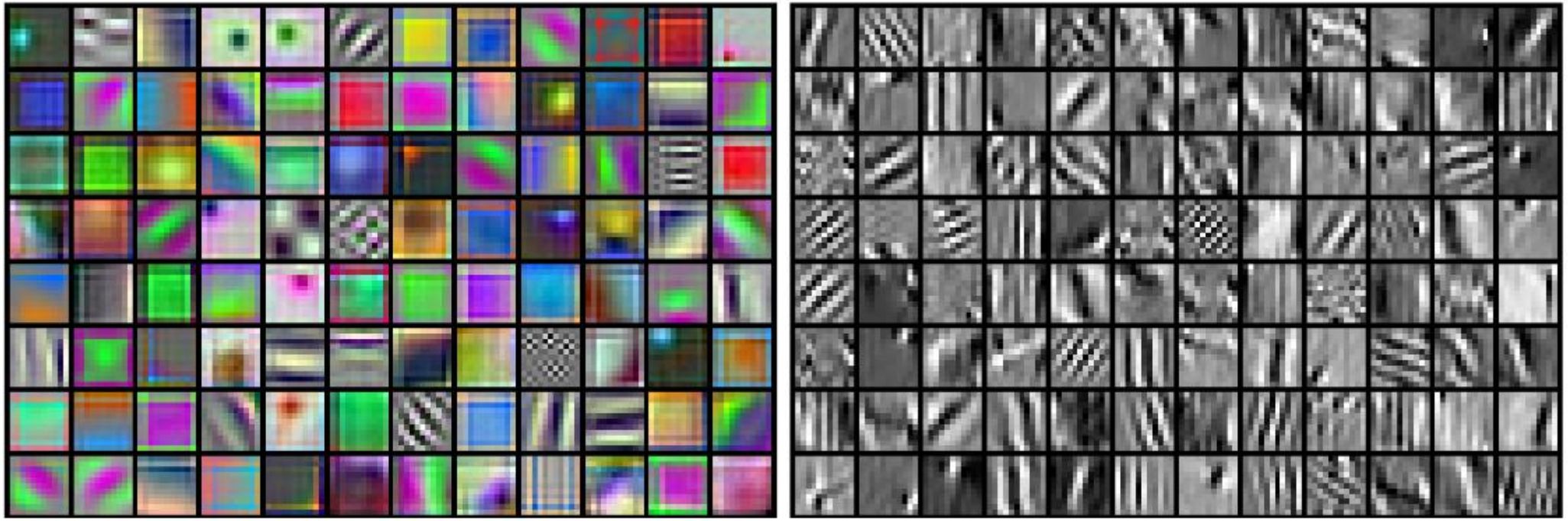


Fig. 3: Filters from the first layer of an AlexNet trained on unsupervised ImageNet on raw RGB input (left) or after a Sobel filtering (right).

Кластеризация (DeepCluster)

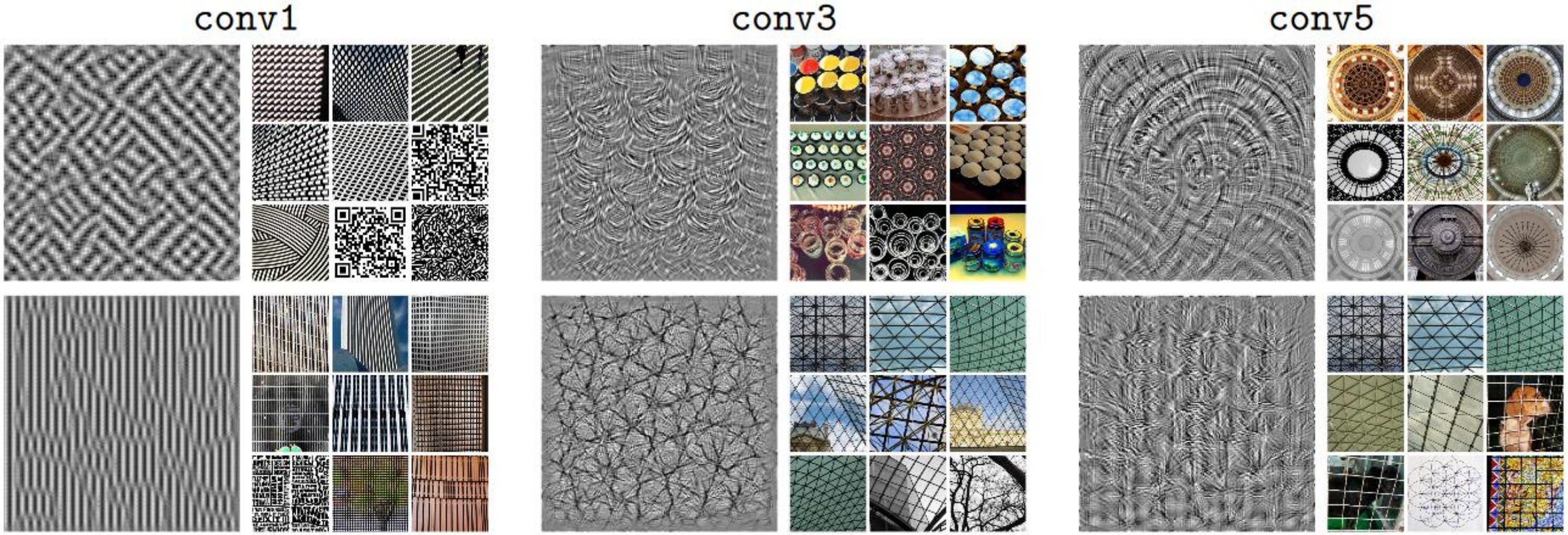


Fig. 4: Filter visualization and top 9 activated images from a subset of 1 million images from YFCC100M for target filters in the layers conv1, conv3 and conv5 of an AlexNet trained with DeepCluster on ImageNet. The filter visualization is obtained by learning an input image that maximizes the response to a target filter [64].

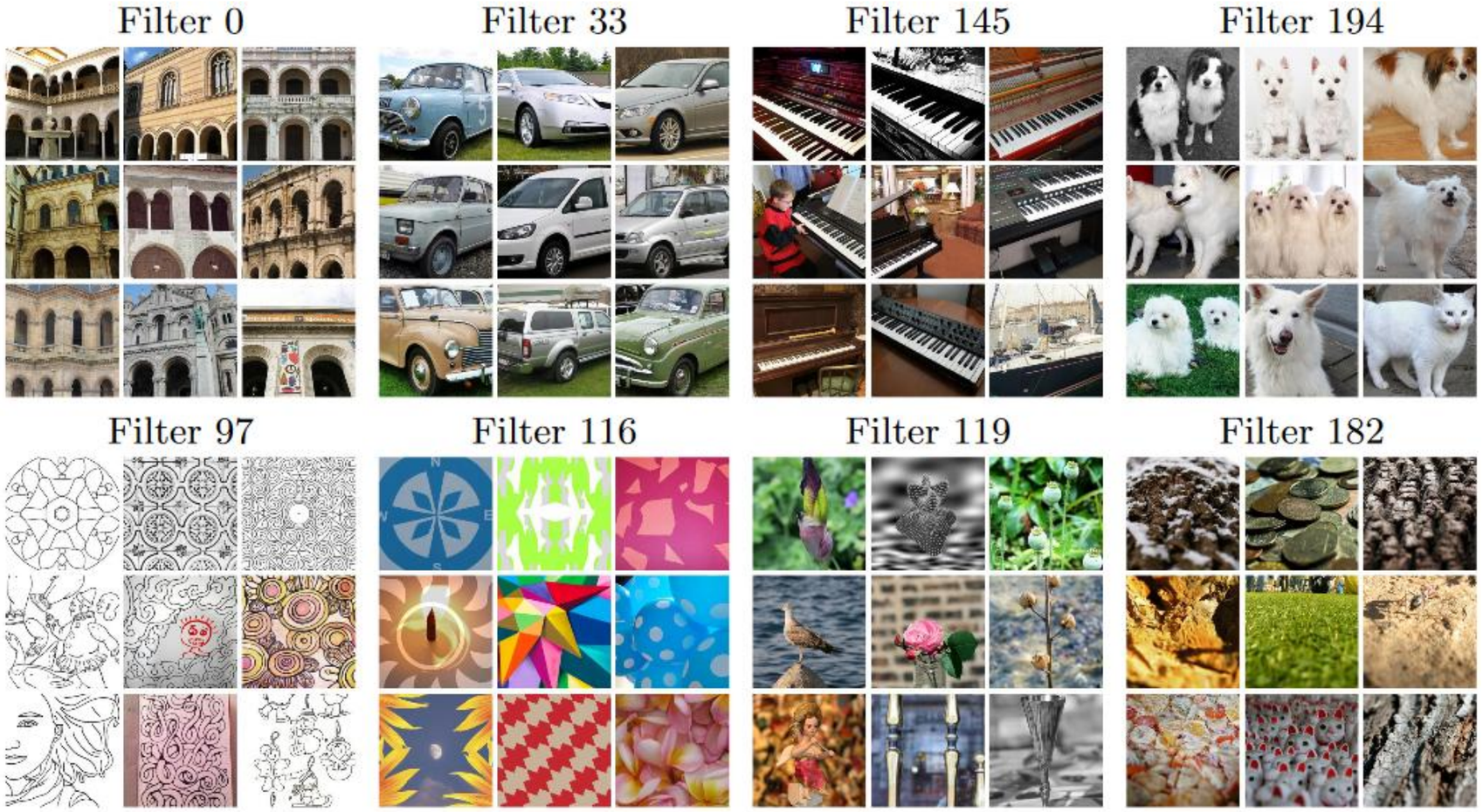


Fig. 5: Top 9 activated images from a random subset of 10 millions images from YFCC100M for target filters in the last convolutional layer. The top row corresponds to filters sensitive to activations by images containing objects. The bottom row exhibits filters more sensitive to stylistic effects. For instance, the filters 119 and 182 seem to be respectively excited by background blur and depth of field effects.

Кластеризация (DeepCluster)

Method	Training set	Classification		Detection		Segmentation	
		FC6-8	ALL	FC6-8	ALL	FC6-8	ALL
Best competitor	ImageNet	63.0	67.7	43.4†	53.2	35.8†	37.7
DeepCluster	ImageNet	72.0	73.7	51.4	55.4	43.2	45.1
DeepCluster	YFCC100M	67.3	69.3	45.6	53.0	39.2	42.2

Table 3: Impact of the training set on the performance of DeepCluster measured on the PASCAL VOC transfer tasks as described in Sec. 4.4. We compare ImageNet with a subset of 1M images from YFCC100M [31]. Regardless of the training set, DeepCluster outperforms the best published numbers on most tasks. Numbers for other methods produced by us are marked with a †

Контекстные кодировщики (Context Encoder) / image inpainting

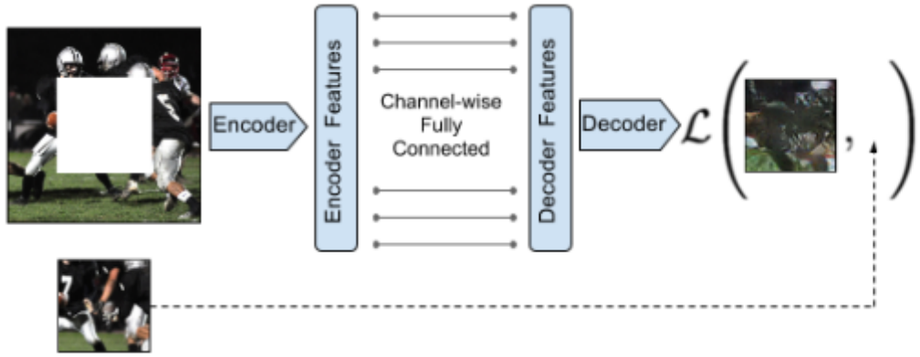
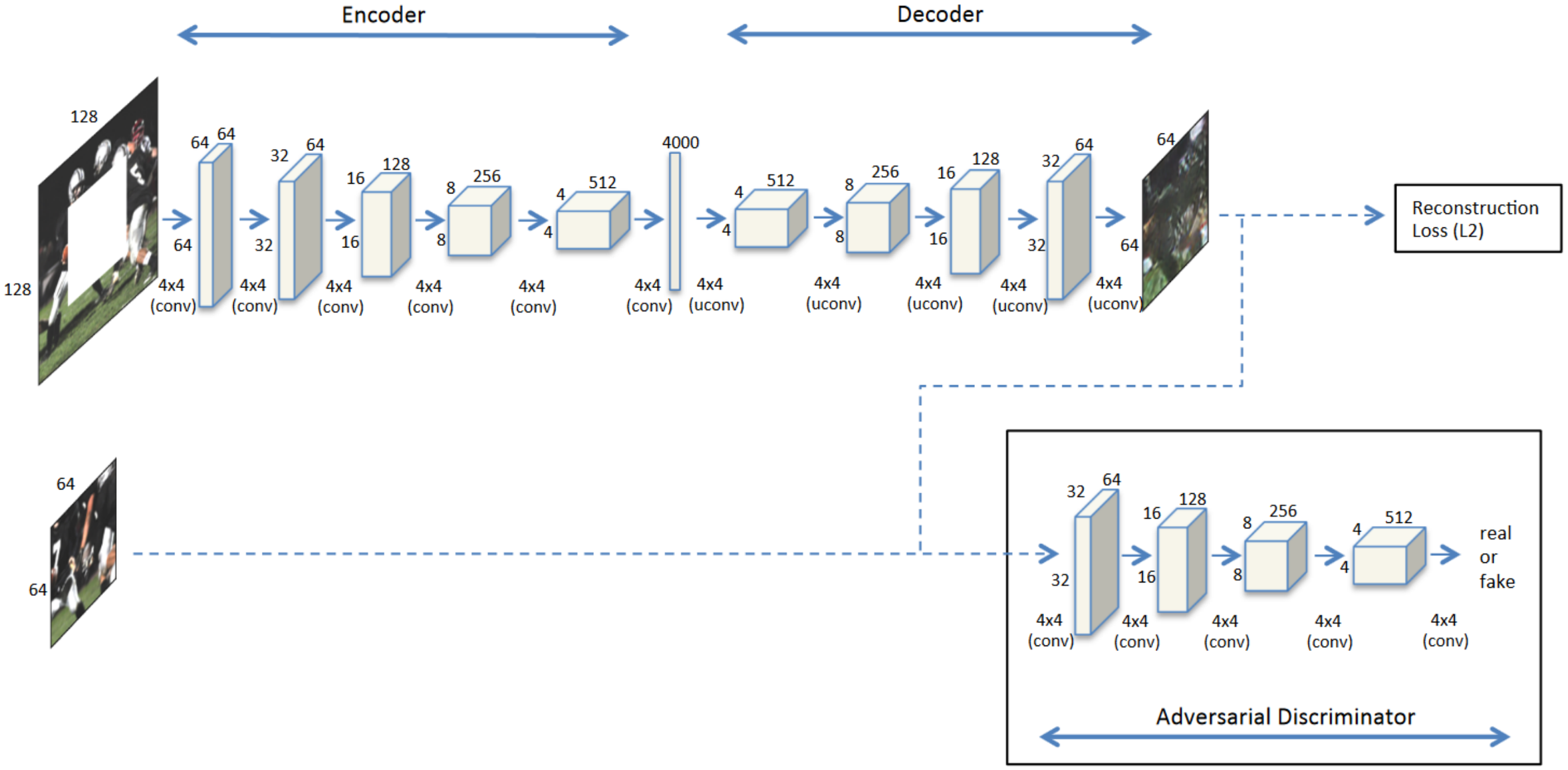


Figure 2: Context Encoder. The context image is passed through the encoder to obtain features which are connected to the decoder using channel-wise fully-connected layer as described in Section 3.1. The decoder then produces the missing regions in the image.

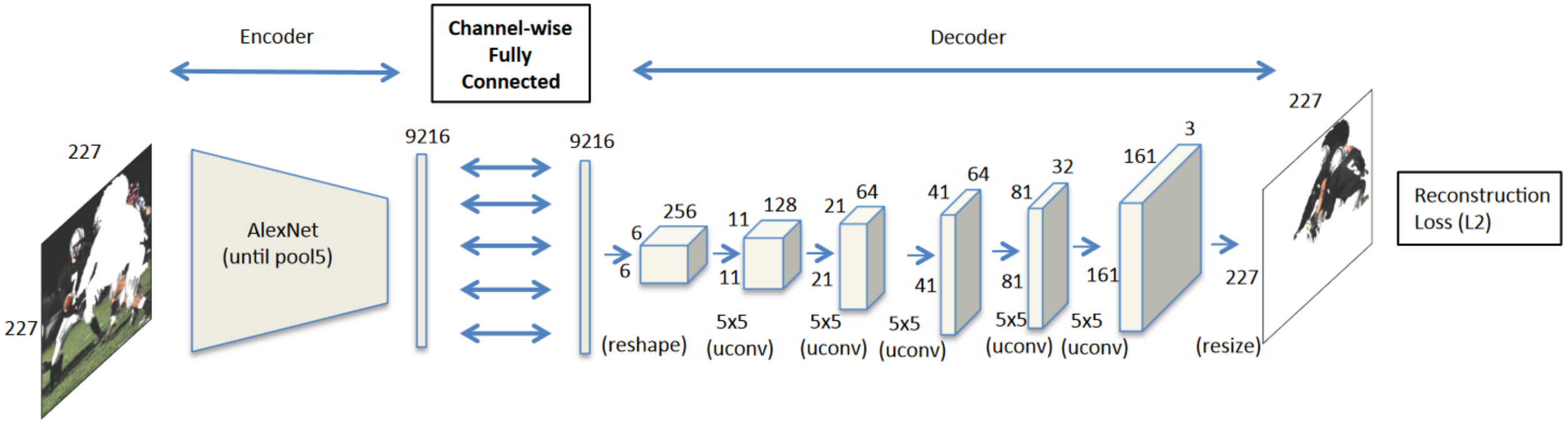
кодировщик ~AlexNet, размер входа = 227×227,
Channel-wise fully-connected layer – специальное соединение «по каналам»
декодировщик – up-convolutional layers

D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. InConference on Computer Vision and Pattern Recognition(CVPR), 2016. https://arxiv.org/abs/1604.07379



(a) Context encoder trained with joint reconstruction and adversarial loss for semantic inpainting. This illustration is shown for *center region dropout*. Similar architecture holds for arbitrary region dropout as well. See Section 3.2.

Контекстные кодировщики (Context Encoder) / image inpainting



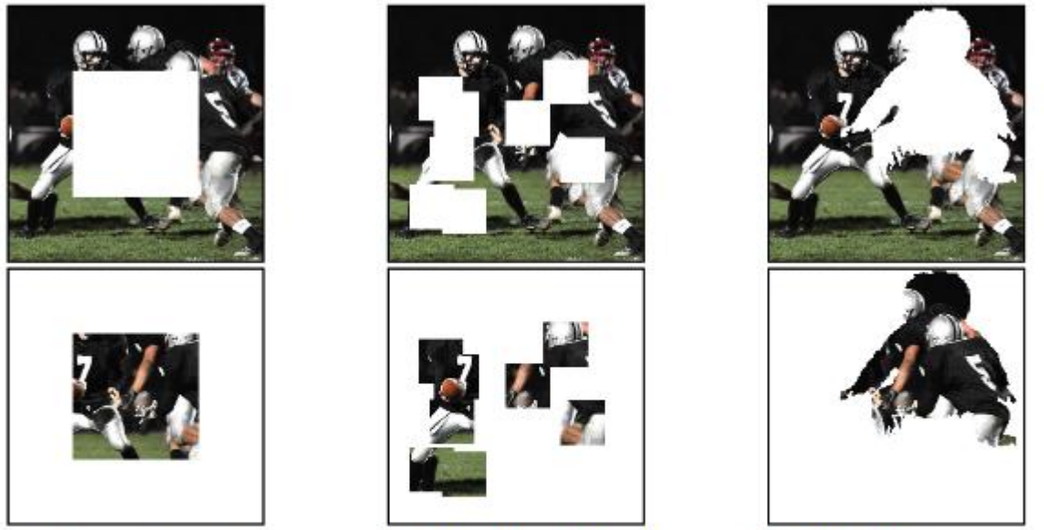
(b) Context encoder trained with reconstruction loss for feature learning by filling in *arbitrary region dropouts* in the input.

Контекстные кодировщики (Context Encoder) / image inpainting

(a) Input context (b) Human artist (c) Context Encoder ($L2$ loss) (d) Context Encoder ($L2$ + Adversarial loss)

Figure 1: Qualitative illustration of the task. Given an image with a missing region (a), a human artist has no trouble inpainting it (b). Automatic inpainting using our *context encoder* trained with $L2$ reconstruction loss is shown in (c), and using both $L2$ and adversarial losses in (d).

Контекстные кодировщики (Context Encoder) / image inpainting



(a) Central region (b) Random block (c) Random region

Figure 3: An example of image x with our different region masks $\hat{M}$ applied, as described in Section 3.3.

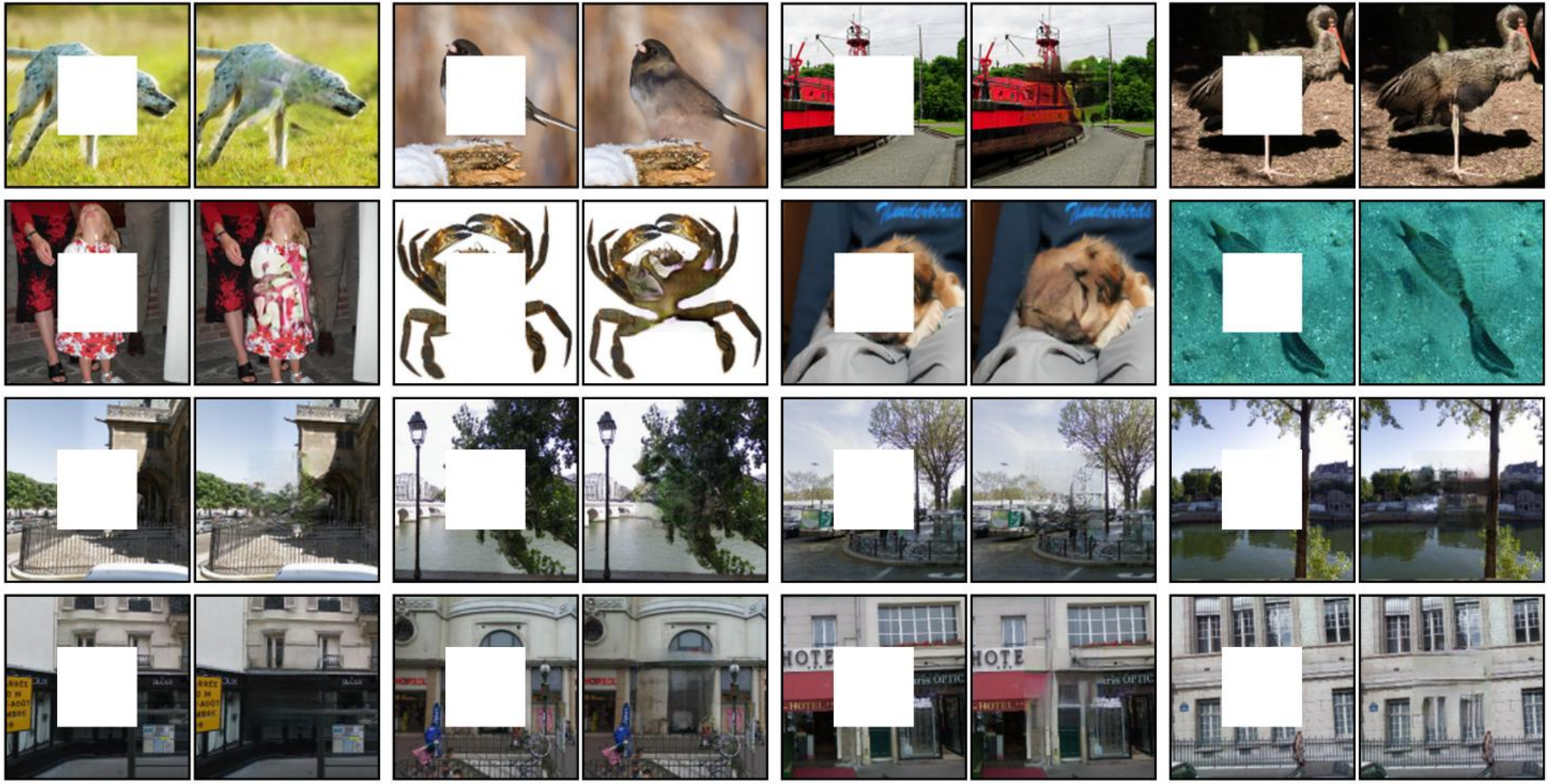


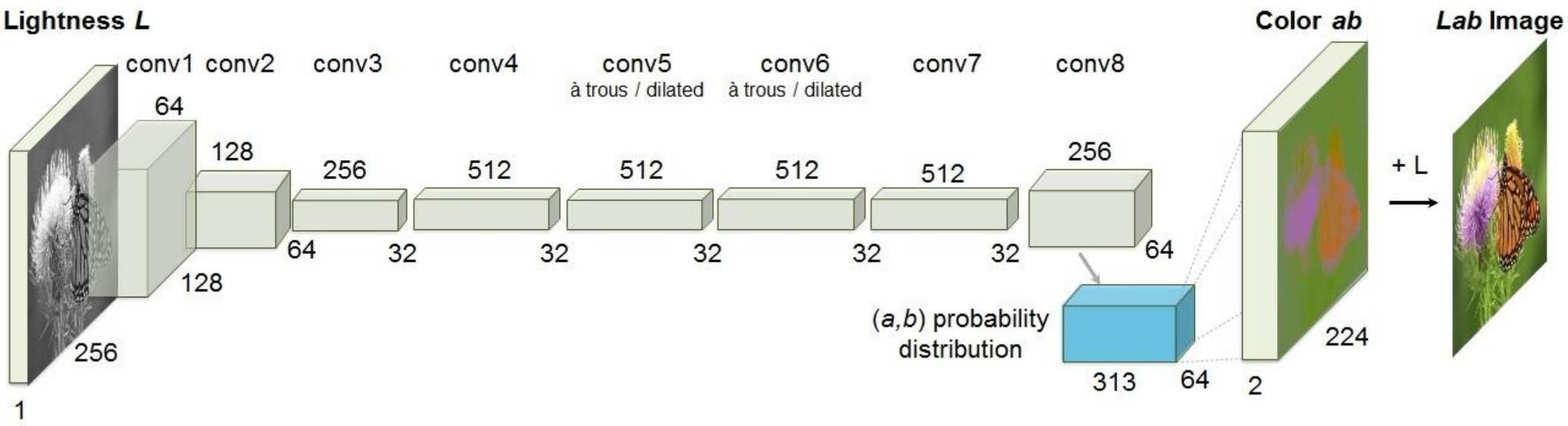
Figure 4: Semantic Inpainting results on *held-out* images for context encoder trained using reconstruction and adversarial loss. First three rows are examples from ImageNet, and bottom two rows are from Paris StreetView Dataset. See more results on author's project website.

Контекстные кодировщики (Context Encoder) / image inpainting

Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	78.2%	56.8%	48.0%
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Wang *et al.* [39]	motion	1 week	58.7%	47.4%	-
Doersch *et al.* [7]	relative context	4 weeks	55.3%	46.6%	-
Ours	context	14 hours	56.5%	44.5%	30.0%

Table 2: Quantitative comparison for classification, detection and semantic segmentation. Classification and Fast-RCNN Detection results are on the PASCAL VOC 2007 test set. Semantic segmentation results are on the PASCAL VOC 2012 validation set from the FCN evaluation described in Section 5.2.3, using the additional training data from [18], and removing overlapping images from the validation set [28].

Раскраска изображений (image colorization)



каждый блок = 2 to 3 conv + ReLu-слои, за ними BN

нет пулинга

Dilated Convolutions

R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In European Conference on Computer Vision (ECCV), 2016, https://richzhang.github.io/colorization/

Fig. 8. Applying our method to legacy black and white photos. Left to right: photo by David Fleay of a Thylacine, now extinct, 1936; photo by Ansel Adams of Yosemite; amateur family photo from 1956; *Migrant Mother* by Dorothea Lange, 1936.

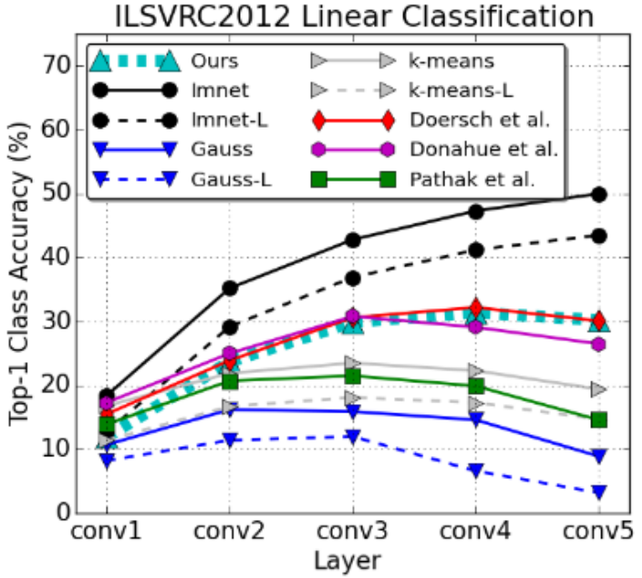


Fig. 7. ImageNet Linear Classification

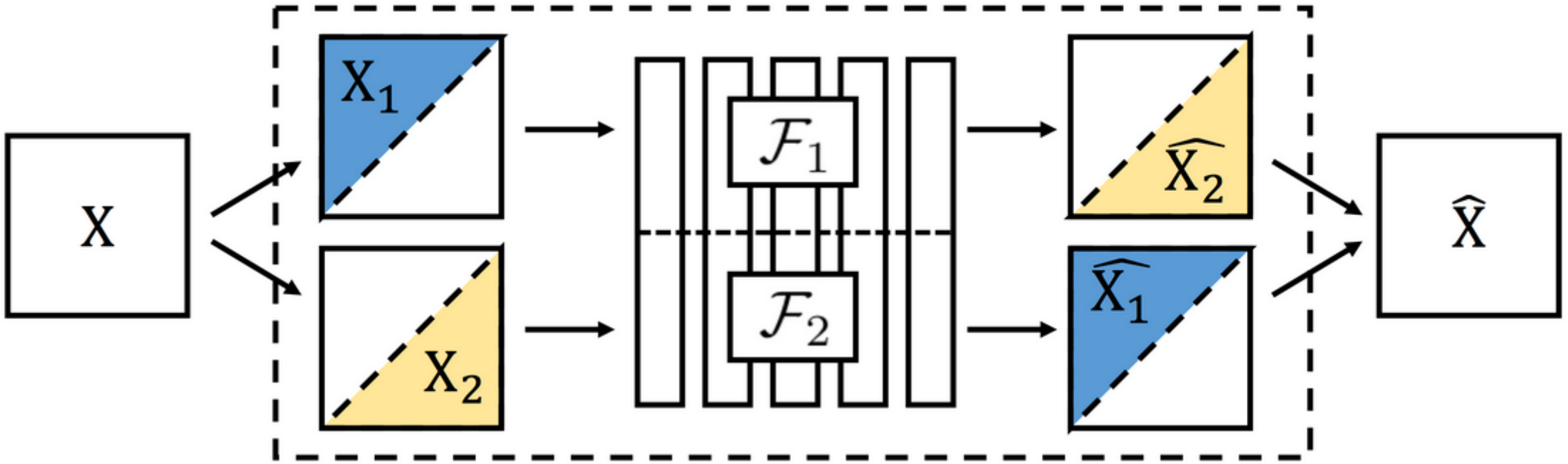
Dataset and Task Generalization on PASCAL [37]								
		Class. (%mAP)			Det. (%mAP)		Seg. (%mIU)	
fine-tune layers	[Ref]	fc8	fc6-8	all	[Ref]	all	[Ref]	all
ImageNet [38]	-	76.8	78.9	79.9	[36]	56.8	[42]	48.0
Gaussian	[10]	–	–	53.3	[10]	43.4	[10]	19.8
Autoencoder	[16]	24.8	16.0	53.8	[10]	41.9	[10]	25.2
k-means [36]	[16]	32.0	39.2	56.6	[36]	45.6	[16]	32.6
Agrawal et al. [8]	[16]	31.2	31.0	54.2	[36]	43.9	–	–
Wang & Gupta [15]	–	28.1	52.2	58.7	[36]	47.4	–	–
*Doersch et al. [14]	[16]	44.7	55.1	**65.3**	[36]	**51.1**	–	–
*Pathak et al. [10]	[10]	–	–	56.5	[10]	44.5	[10]	29.7
*Donahue et al. [16]	–	38.2	50.2	58.6	[16]	46.2	[16]	34.9
Ours (gray)	–	**52.4**	**61.5**	**65.9**	–	46.1	–	35.0
Ours (color)	–	**52.4**	**61.5**	**65.6**	–	46.9	–	**35.6**

Table 2. PASCAL Tests

Fig. 7. Task Generalization on ImageNet We freeze pre-trained networks and learn linear classifiers on internal layers for ImageNet [28] classification. Features are average-pooled, with equal kernel and stride sizes, until feature dimensionality is below 10k. ImageNet [38], k-means [36], and Gaussian initializations were run with grayscale inputs, shown with dotted lines, as well as color inputs, shown with solid lines. Previous [14,10] and concurrent [16] self-supervision methods are shown.

Tab. 2. Task and Dataset Generalization on PASCAL Classification and detection on PASCAL VOC 2007 [39] and segmentation on PASCAL VOC 2012 [40], using standard mean average precision (mAP) and mean intersection over union (mIU) metrics for each task. We fine-tune our network with grayscale inputs (gray) and color inputs (color). Methods noted with a * only pre-trained a subset of the AlexNet layers. The remaining layers were initialized with [36]. Column **Ref** indicates the source for a value obtained from a previous paper.

Расщеплённые автокодировщики (Split-brain autoencoders)



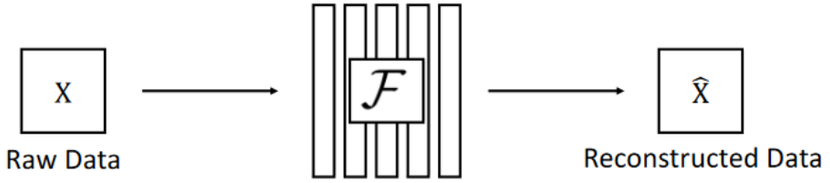
каналы разбиваем на 2 группы

две сети обучаются по одной группе предсказывать другую

R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. InConference on Computer Vision and Pattern Recognition(CVPR), 2017. https://arxiv.org/abs/1611.09842

Расщеплённые автокодировщики (Split-brain autoencoders)

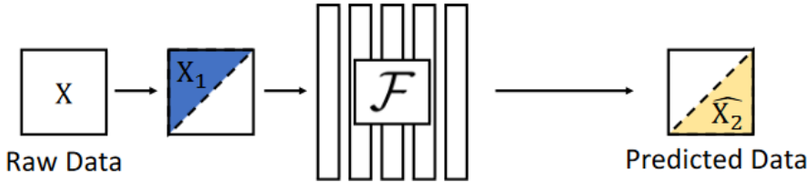
Traditional Autoencoder



Реконструкция сырых данных

Узкое горло

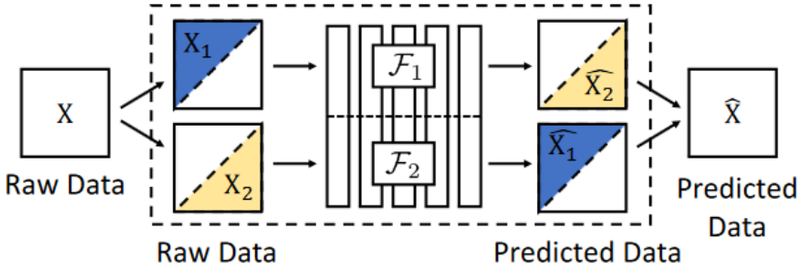
Cross-Channel Encoder



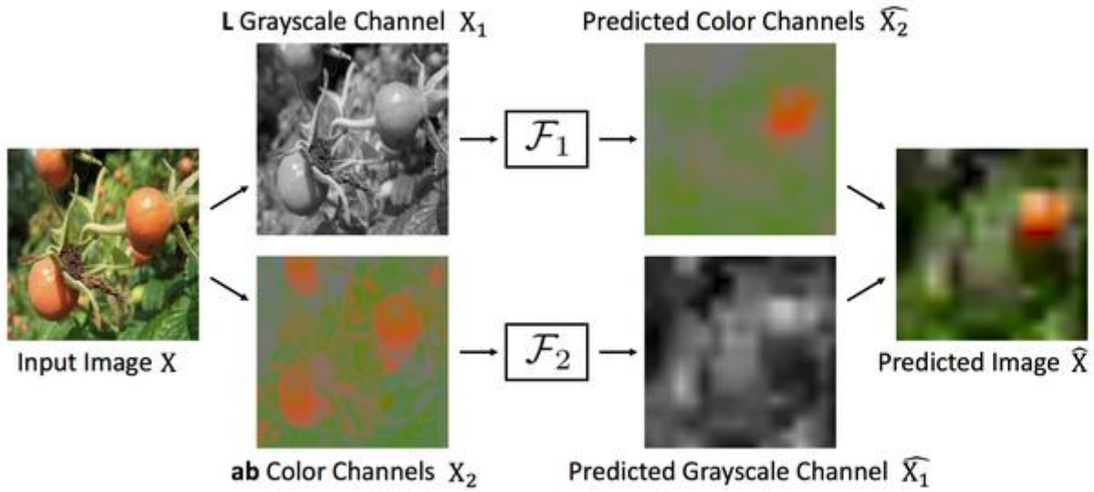
Восстановление каналов (по другим)

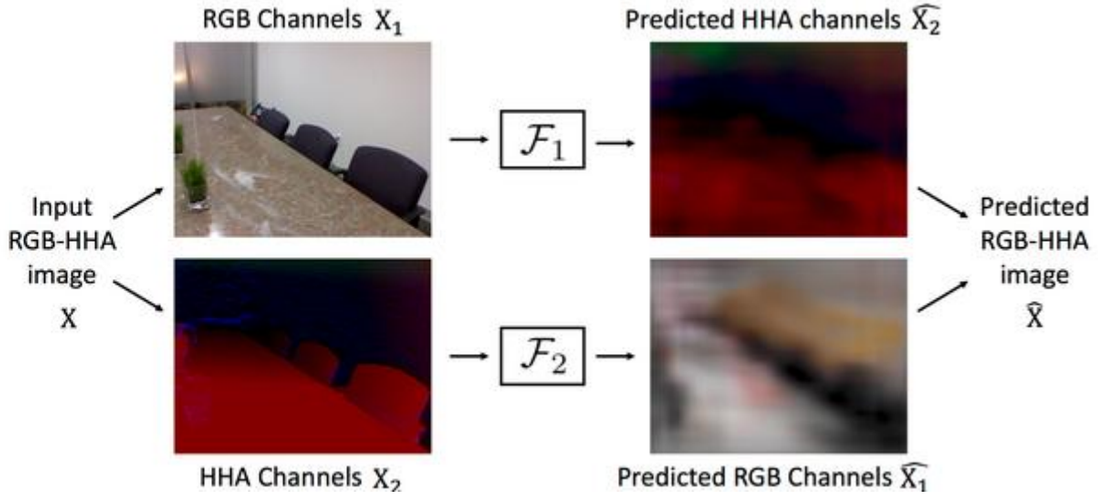
Неплохие признаки

Split-Brain Autoencoder



Совсем хорошие признаки





Сегментация, порождённая движением (motion segmentation prediction)

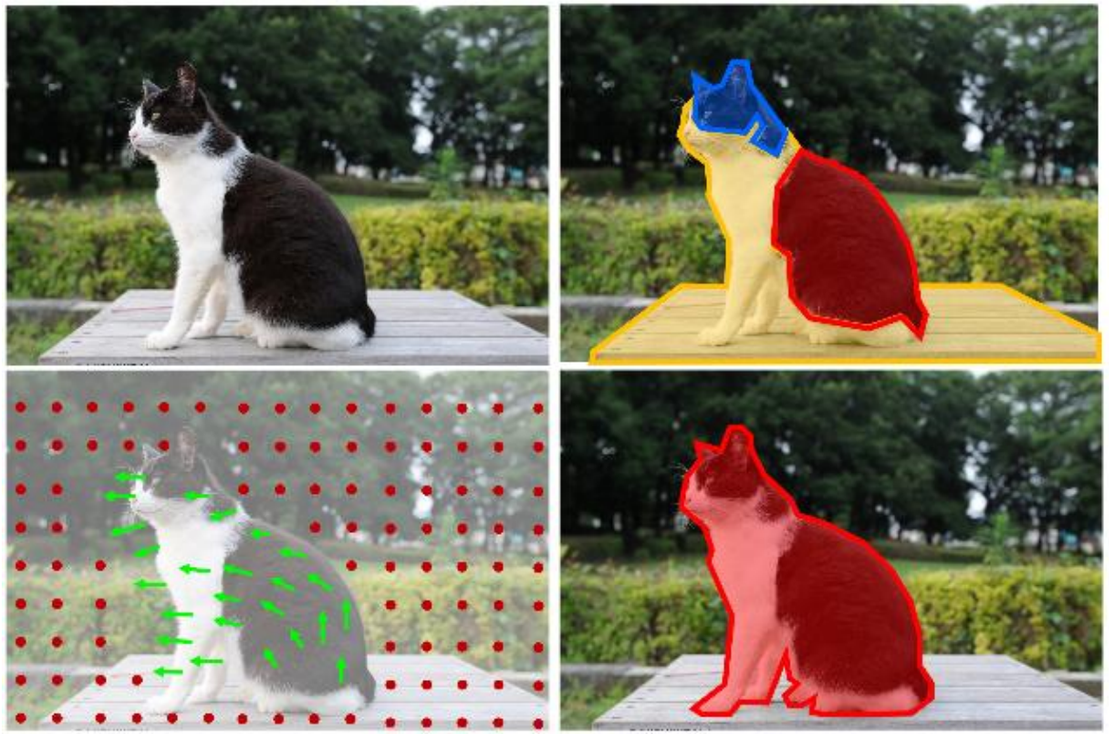


Figure 1. Low-level appearance cues lead to incorrect grouping (top right). Motion helps us to correctly group pixels that move together (bottom left) and identify this group as a single object (bottom right). We use unsupervised motion-based grouping to train a ConvNet to segment objects in *static images* and show that the network learns strong features that transfer well to other tasks.

на видео пиксели, которые перемещаются вместе, скорее всего, принадлежат одному объекту

– так получаем разметку,
а её уже восстанавливаем из статичного кадра

D. Pathak, R. B. Girshick, P. Dollar, T. Darrell, and B. Hariharan. Learning features by watching objects move. In Conference on Computer Vision and Pattern Recognition(CVPR), 2017 https://arxiv.org/abs/1612.06370

Сегментация, порождённая движением (motion segmentation prediction)

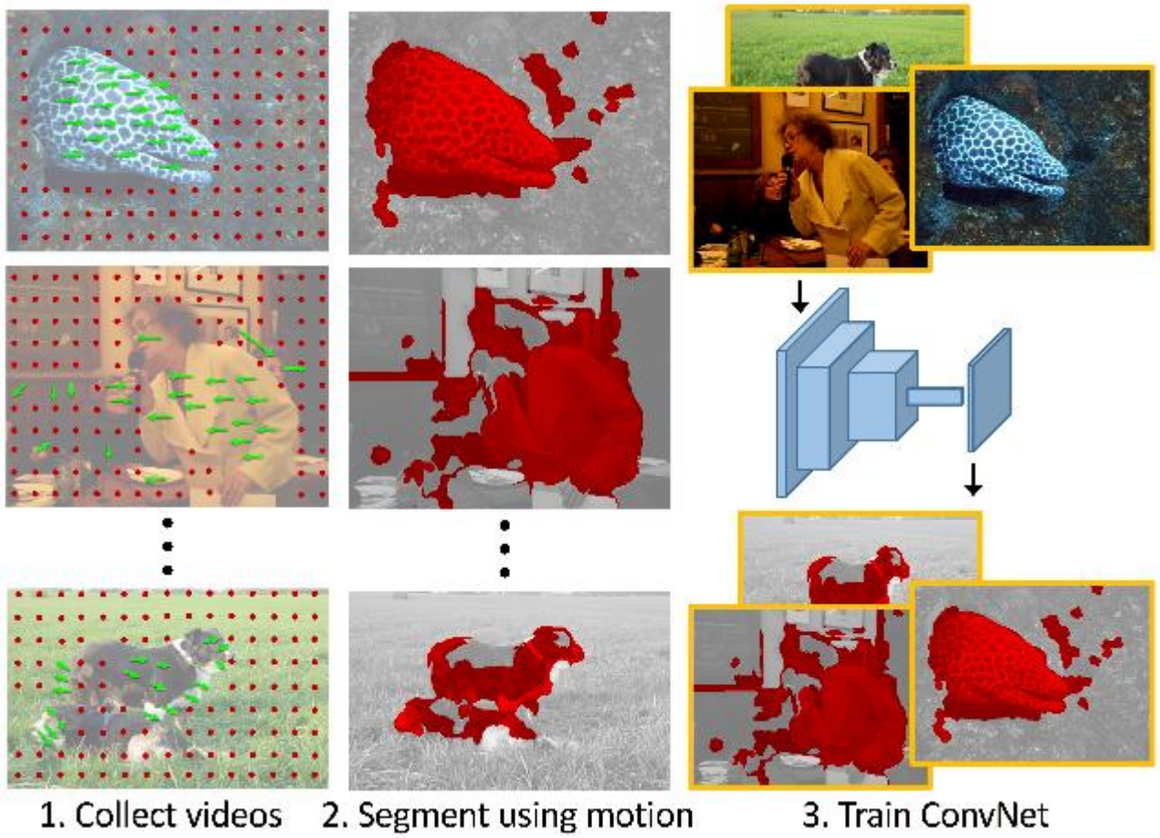


Figure 2. Overview of our approach. We use motion cues to segment objects in videos *without any supervision*. We then train a ConvNet to predict these segmentations from *static frames*, *i.e.* without any motion cues. We then transfer the learned representation to other recognition tasks.

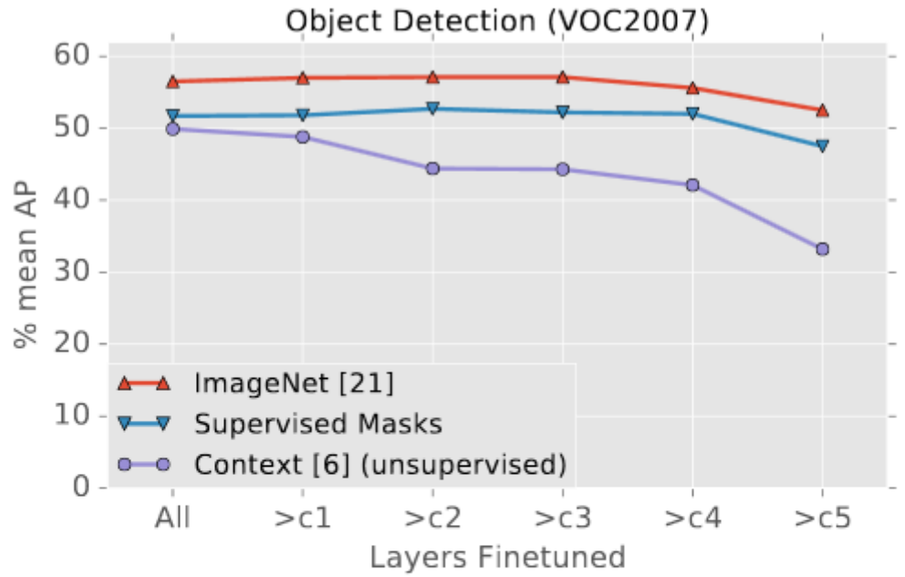


Figure 3. Our representation trained on manually-annotated segments from COCO (without class labels) compared to ImageNet pretraining and context prediction (unsupervised) [8], evaluated for object detection on PASCAL VOC 2007. '>cX': all layers above convX are fine-tuned; 'All': the entire net is fine-tuned.

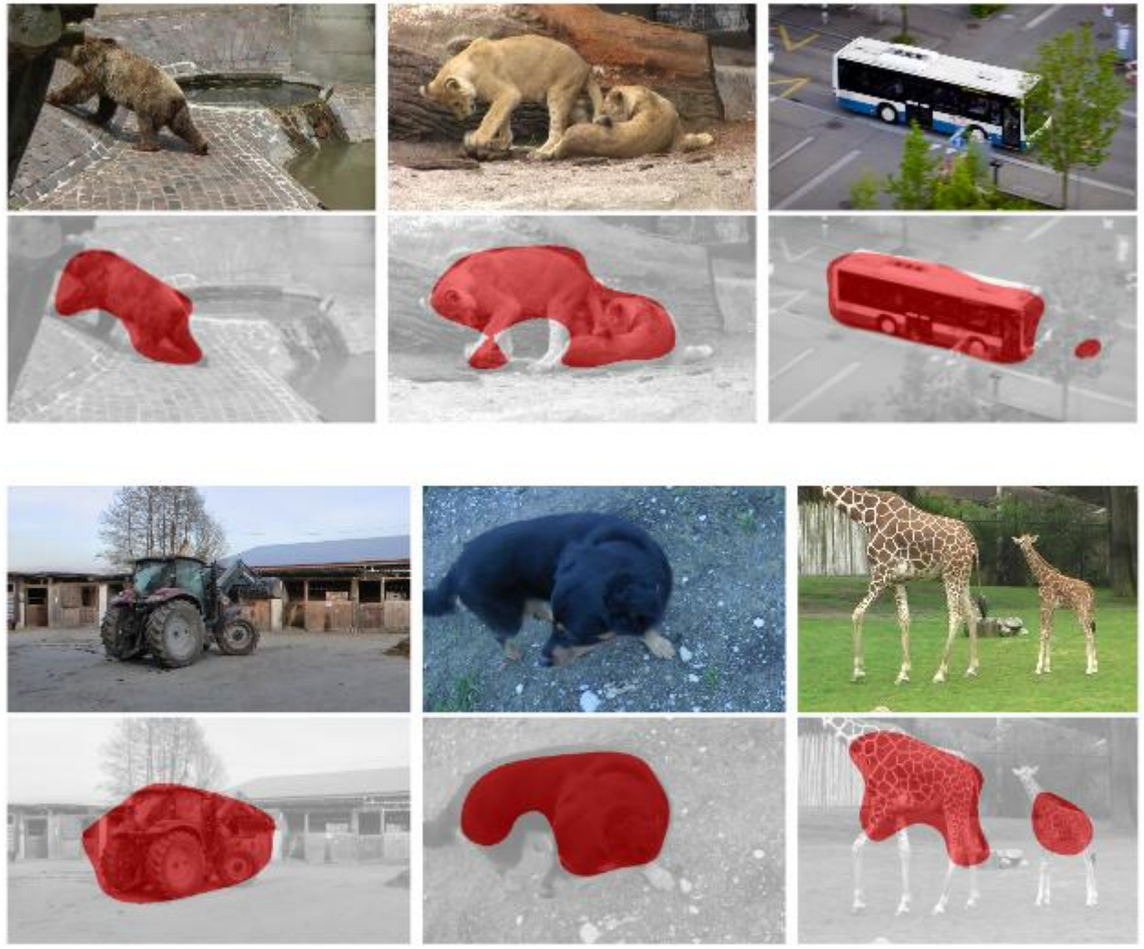


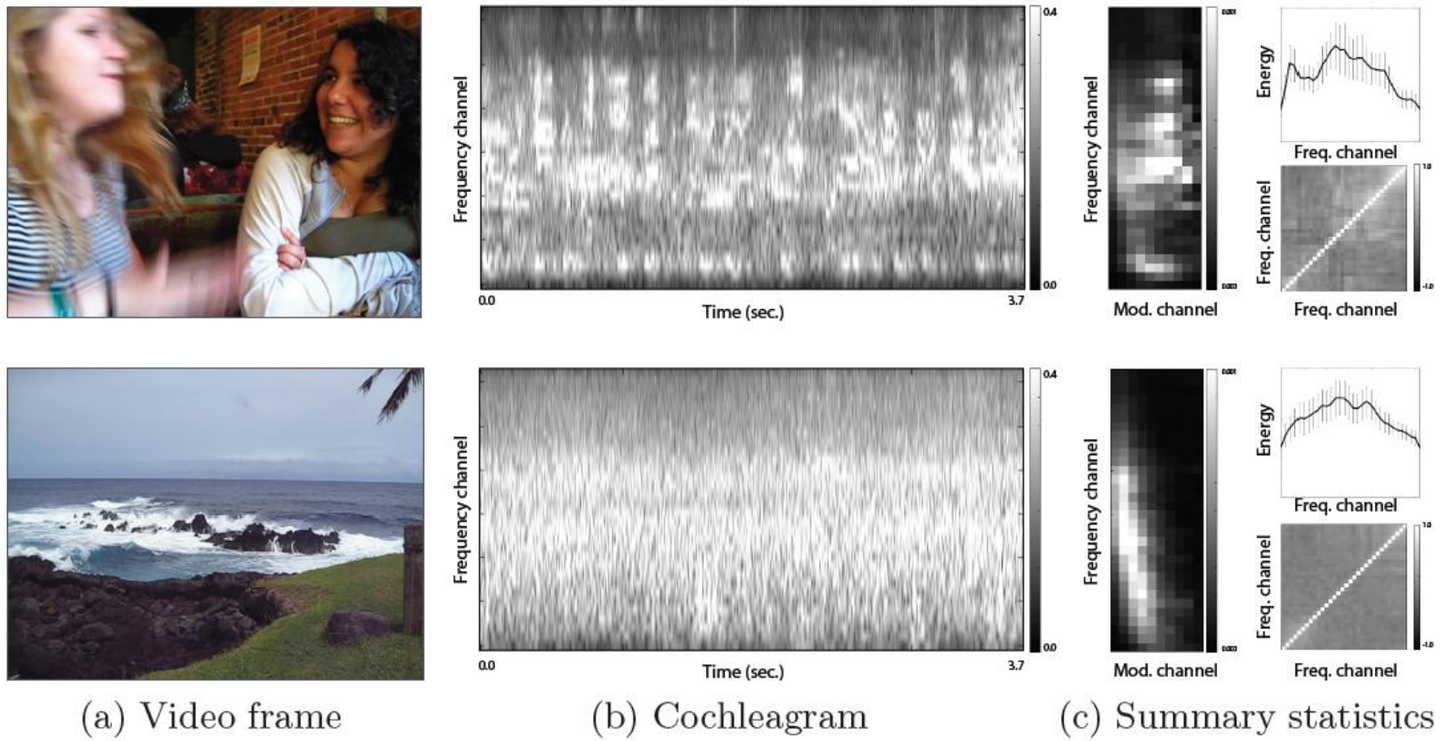
Figure 7. Examples of segmentations produced by our ConvNet on held out images. The ConvNet is able to identify the motile object (or objects) and segment it out from a single frame. Masks are not perfect but they do capture the general object shape.

лучше SOTA в USL, но чуть хуже предтренировки на размеченных данных

	Full train set						150 image set						
Method	All	>c1	>c2	>c3	>c4	>c5	All	>c1	>c2	>c3	>c4	>c5	#wins
Supervised													
Imagenet	56.5	57.0	57.1	57.1	55.6	52.5	17.7	19.1	19.7	20.3	20.9	19.6	NA
Sup. Masks (Ours)	51.7	51.8	52.7	52.2	52.0	47.5	13.6	13.8	15.5	17.6	18.1	15.1	NA
Unsupervised													
Jigsaw‡ [30]	49.0	50.0	48.9	47.7	45.8	37.1	5.9	8.7	8.8	10.1	9.9	7.9	NA
Kmeans [23]	42.8	42.2	40.3	37.1	32.4	26.0	4.1	4.9	5.0	4.5	4.2	4.0	0
Egomotion [2]	37.4	36.9	34.4	28.9	24.1	17.1	–	–	–	–	–	–	0
Inpainting [35]	39.1	36.4	34.1	29.4	24.8	13.4	–	–	–	–	–	–	0
Tracking-gray [46]	43.5	44.6	44.6	44.2	41.5	35.7	3.7	5.7	7.4	9.0	9.4	9.0	0
Sounds [33]	42.9	42.3	40.6	37.1	32.0	26.5	5.4	5.1	5.0	4.8	4.0	3.5	0
BiGAN [10]	44.9	44.6	44.7	42.4	38.4	29.4	4.9	6.1	7.3	7.6	7.1	4.6	0
Colorization [51]	44.5	44.9	44.7	44.4	42.6	38.0	6.1	7.9	8.6	10.6	10.7	9.9	0
Split-Brain Auto [52]	43.8	45.6	45.6	46.1	44.1	37.6	3.5	7.9	9.6	10.2	11.0	10.0	0
Context [8]	**49.9**	**48.8**	44.4	44.3	42.1	33.2	6.7	**10.2**	9.2	9.5	9.4	8.7	3
Context-videos† [8]	47.8	47.9	46.6	**47.2**	44.3	33.4	6.6	9.2	10.7	12.2	11.2	9.0	1
Motion Masks (Ours)	48.6	48.2	**48.3**	47.0	**45.8**	**40.3**	**10.2**	**10.2**	**11.7**	**12.5**	**13.3**	**11.0**	**9**

Table 1. Object detection AP (%) on PASCAL VOC 2012 using Fast R-CNN with various pretrained ConvNets. All models are trained on `train` and tested on `val` using consistent Fast R-CNN settings. '–' means training didn't converge due to insufficient data. Our approach achieves the best performance in the majority of settings. †Doersch *et al.* [8] trained their original context model using ImageNet images. The Context-videos model is obtained by retraining their approach on our video frames from YFCC. This experiment controls for the effect of the distribution of training images and shows that the image domain used for training does not significantly impact performance. ‡Noroozi *et al.* [30] use a more computationally intensive ConvNet architecture (>2× longer to finetune) with a finer stride at conv1, preventing apples-to-apples comparisons. Nevertheless, their model works significantly worse than our representation when either layers are frozen or in case of limited data and is comparable to ours when network is finetuned with full training data.

Разметка окружающими звуками (ambient sounds)



(a) Video frame (b) Cochleagram (c) Summary statistics

Fig. 1: Visual scenes are associated with characteristic sounds. Our goal is to take an image (a) and predict time-averaged summary statistics (c) of a cochleagram (b). The statistics we use are (clockwise): the response to a bank of band-pass modulation filters; the mean and standard deviation of each frequency band; and the correlation between bands. We show two frames from the Flickr video dataset [34]. The first contains the sound of human speech; the second contains the sound of wind and crashing waves. The differences between these sounds are reflected in their summary statistics: e.g., the water/wind sound, which is similar to white noise, contains fewer correlations between cochlear channels.

Разметка окружающими звуками (ambient sounds)

по кадрам видео-изображения с помощью CNN предсказываем статистики звуков
усредняются за несколько секунд
т.к. звук не всегда привязан к картинке

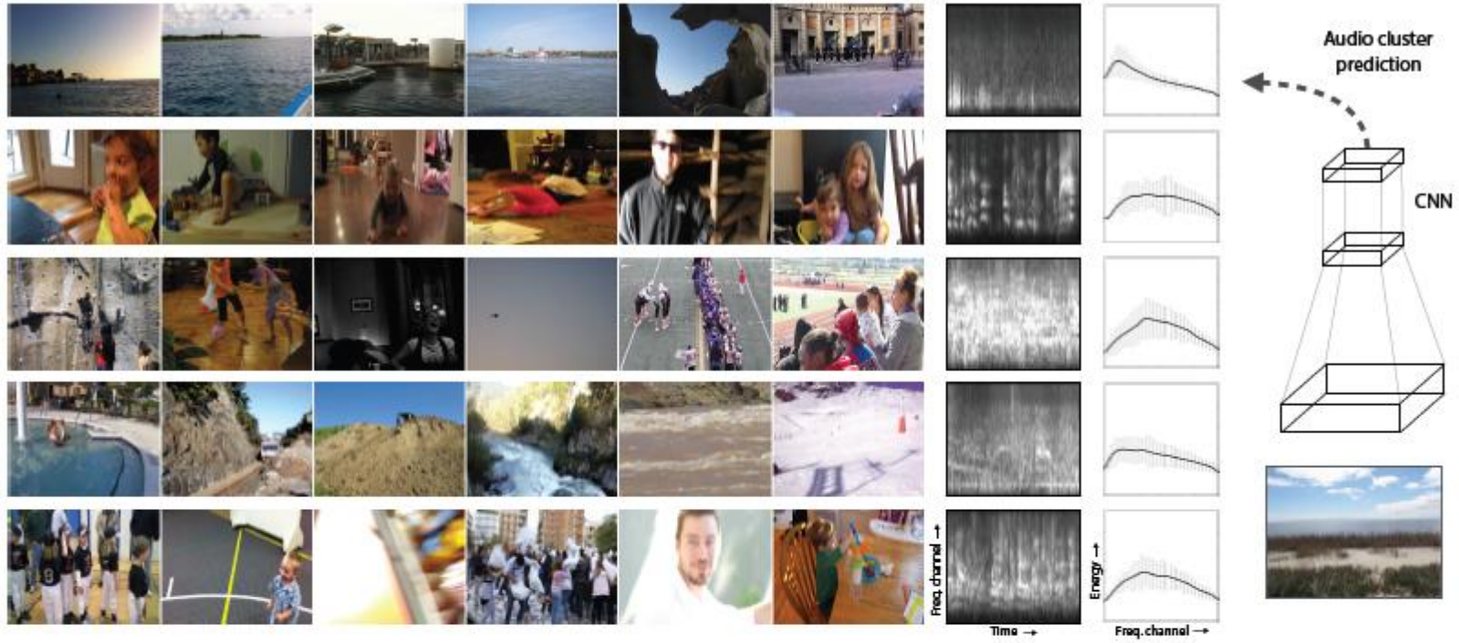
- the mean and standard deviation of each frequency channel
- the mean squared response of each of a bank of modulation lters applied to each channel
- the Pearson correlation between pairs of channels

Был также кластерный формат –
статистики кластеризуются с помощью k-means (см. ниже)

Ещё вариант – PCA + бинаризация нескольких главных компонент
так порождаем метки для обучения

A. Owens, J. Wu, J. H. McDermott, W. T. Freeman, and A. Torralba. Ambient sound provides supervision for visuallearning.ECCV, 2016 https://arxiv.org/abs/1608.07017

Разметка окружающими звуками (ambient sounds)



(a) Images grouped by audio cluster (b) Clustered audio stats. (c) CNN model

Fig. 2: Visualization of some of the audio clusters used in one of our models (5 of 30 clusters). For each cluster, we show (a) the images in the test set whose sound textures were closest to the centroid (no more than one frame per video), and (b) we visualize aspects of the sound texture used to define the cluster centroid – specifically, the mean and standard deviation of the frequency channels. We also include a representative cochleagram (that of the leftmost image). Although the clusters were defined using audio, there are common objects and scene attributes in many of the images. We train a CNN to predict a video frame's auditory cluster assignment (c).

Подсчёт примитивов (counting visual primitives)

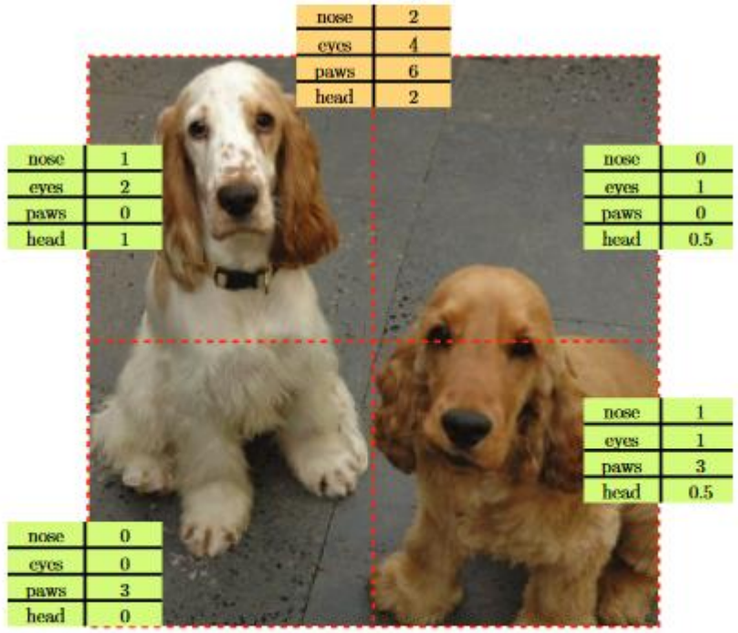


Figure 1: The number of visual primitives in the whole image should match the sum of the number of visual primitives in each tile (dashed red boxes).

Центральные идеи:

1) если делить изображение на куски, то число примитивов = сумме чисел в кусках

2) на другом изображении, скорее всего, другое число примитивов

3) если делать какое-то преобразование (ex: масштабирование), то число примитивов сохраняется

– решается как задача регрессии

Какие именно примитивы считаются – непонятно

M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. InInternational Conferenceon Computer Vision (ICCV), 2017. https://arxiv.org/abs/1708.06734

Подсчёт примитивов (counting visual primitives)

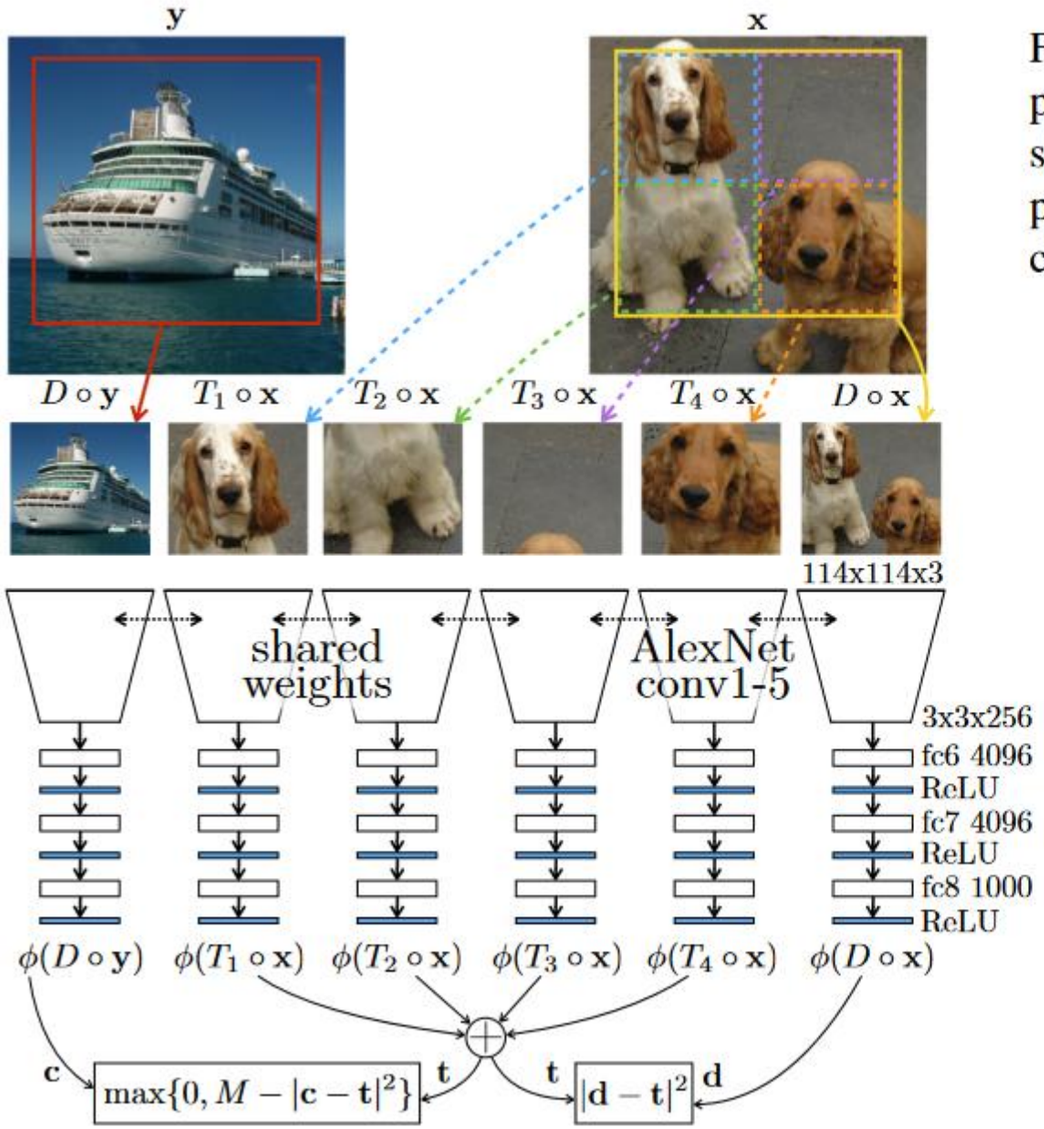


Figure 2: **Training AlexNet to learn to count.** The proposed architecture uses a siamese arrangement so that we simultaneously produce features for 4 tiles and a downsampled image. We also compute the feature from a randomly chosen downsampled image ($D \circ \mathbf{y}$) as a contrastive term.

Подсчёт примитивов (counting visual primitives)

Method	Ref	Class.	Det.	Segm.
Supervised [20]	[43]	79.9	56.8	48.0
Random	[33]	53.3	43.4	19.8
Context [9]	[19]	55.3	46.6	-
Context [9]*	[19]	65.3	51.1	-
Jigsaw [30]	[30]	67.6	**53.2**	37.6
ego-motion [1]	[1]	52.9	41.8	-
ego-motion [1]*	[1]	54.2	43.9	-
Adversarial [10]*	[10]	58.6	46.2	34.9
ContextEncoder [33]	[33]	56.5	44.5	29.7
Sound [31]	[44]	54.4	44.0	-
Sound [31]*	[44]	61.3	-	-
Video [41]	[19]	62.8	47.4	-
Video [41]*	[19]	63.1	47.2	-
Colorization [43]*	[43]	65.9	46.9	35.6
Split-Brain [44]*	[44]	67.1	46.7	36.0
ColorProxy [22]	[22]	65.9	-	**38.0**
WatchingObjectsMove [32]	[32]	61.0	52.2	-
Counting		**67.7**	51.4	36.6

Table 1: **Evaluation of transfer learning on PASCAL.** Classification and detection are evaluated on PASCAL VOC 2007 in the frameworks introduced in [19] and [11] respectively. Both tasks are evaluated using mean average precision (mAP) as a performance measure. Segmentation is evaluated on PASCAL VOC 2012 in the framework of [26], which reports mean intersection over union (mIoU). (*) denotes the use of the data initialization method [19].

Подсчёт примитивов (counting visual primitives)

Method	conv1	conv2	conv3	conv4	conv5
Supervised [20]	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Context [9]	16.2	23.3	30.2	31.7	29.6
Jigsaw [30]	**18.2**	28.8	34.0	33.9	27.1
ContextEncoder [33]	14.1	20.7	21.0	19.8	15.5
Adversarial [10]	17.7	24.5	31.0	29.9	28.0
Colorization [43]	12.5	24.5	30.4	31.5	30.3
Split-Brain [44]	17.7	29.3	**35.4**	**35.2**	**32.8**
Counting	18.0	**30.6**	34.3	32.5	25.7

Table 2: **ImageNet classification with a linear classifier.** We use the publicly available code and configuration of [43]. Every column shows the top-1 accuracy of AlexNet on the classification task. The learned weights from `conv1` up to the displayed layer are frozen. The features of each layer are spatially resized until there are fewer than 9K dimensions left. A fully connected layer followed by softmax is trained on a 1000-way object classification task.

Method	conv1	conv2	conv3	conv4	conv5
Places labels [45]	22.1	35.1	40.2	43.3	44.6
ImageNet labels [20]	22.7	34.8	38.4	39.4	38.7
Random	15.7	20.3	19.8	19.1	17.5
Context [9]	19.7	26.7	31.9	32.7	30.9
Jigsaw [30]	23.0	31.9	35.0	34.2	29.3
Context encoder [33]	18.2	23.2	23.4	21.9	18.4
Sound [31]	19.9	29.3	32.1	28.8	29.8
Adversarial [10]	22.0	28.7	31.8	31.3	29.7
Colorization [43]	16.0	25.7	29.6	30.3	29.7
Split-Brain [44]	21.3	30.7	34.0	34.1	**32.5**
Counting	**23.3**	**33.9**	**36.3**	**34.7**	29.6

Table 3: **Places classification with a linear classifier.** We use the same setting as in Table 2 except that to evaluate generalization across datasets, the model is pretrained on ImageNet (with no labels) and then tested with frozen layers on Places (with labels). The last layer has 205 neurons for scene categories.

Подсчёт примитивов (counting visual primitives)

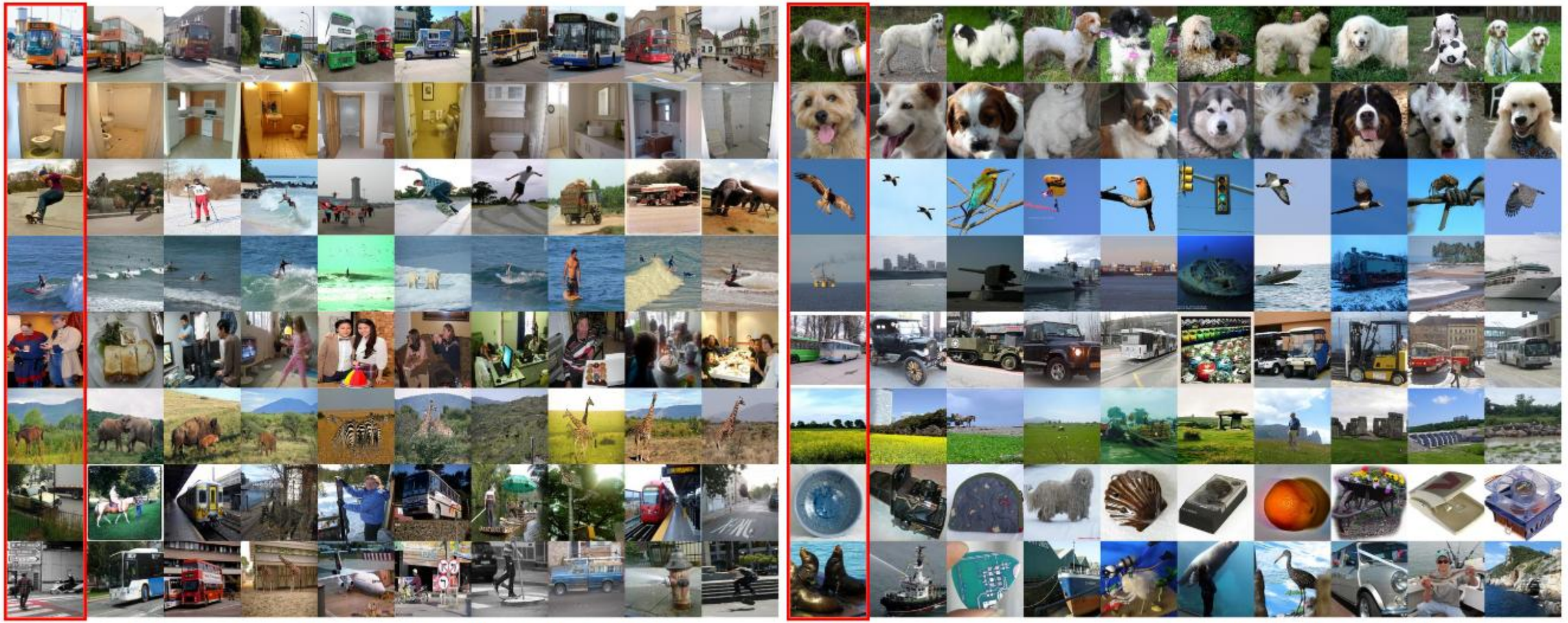


Figure 7: **Nearest neighbor retrievals.** Left: COCO retrievals. Right: ImageNet retrievals. In both datasets, the leftmost column (with a red border) shows the queries and the other columns show the top matching images sorted with increasing Euclidean distance in our counting feature space from left to right. On the bottom 3 rows, we show the failure retrieval cases. Note that the matches share a similar content and scene outline.

Multi-task Self-Supervised Visual Learning

Self-Supervised Tasks

1) Relative Position

для соседних кропов предсказать
up, down, left, right, left-up, right-up, left-down, right-down
чтобы не было обучения под цвет 2 цветовых канала заменяются шумом

2) Colorization

3) Exemplar

triplet loss для отличия патчей из одного изображения и из разных

4) Motion Segmentation

в видео предсказать, какие пиксели будут в следующем фрейме

Carl Doersch, Andrew Zisserman «Multi-task Self-Supervised Visual Learning» //
https://arxiv.org/pdf/1708.07860.pdf

Multi-task Self-Supervised Visual Learning

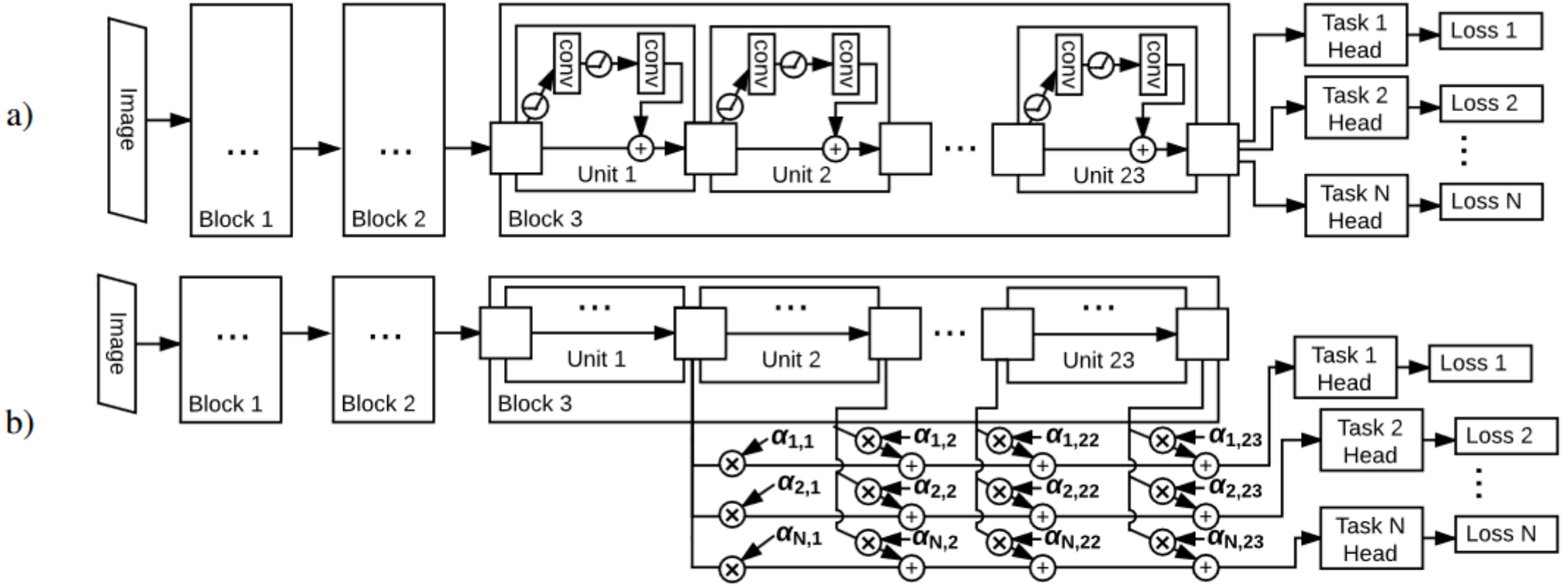


Figure 1. The structure of our multi-task network. It is based on ResNet-101, with block 3 having 23 residual units. a) Naive shared-trunk approach, where each "head" is attached to the output of block 3. b) the lasso architecture, where each "head" receives a linear combination of unit outputs within block3, weighted by the matrix α, which is trained to be sparse.

удалён 4 блок из ResNet-101

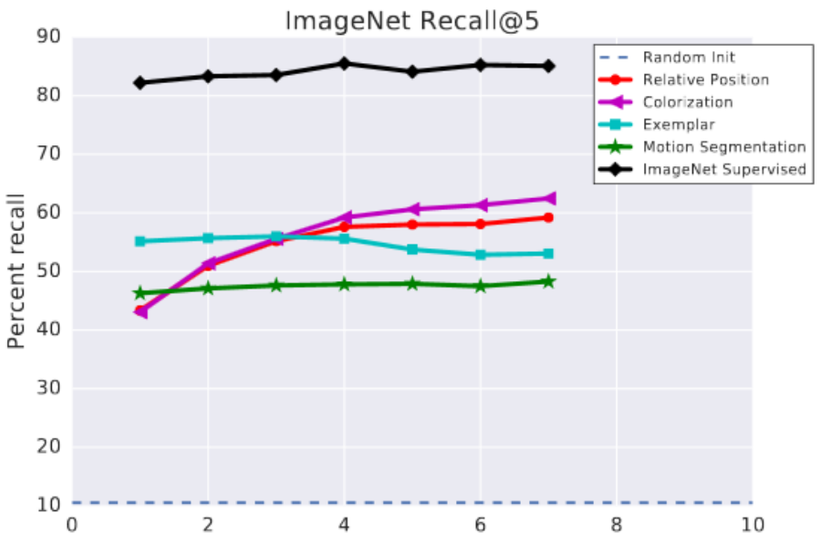
L1-ошибка для разреженности (маленький эффект)

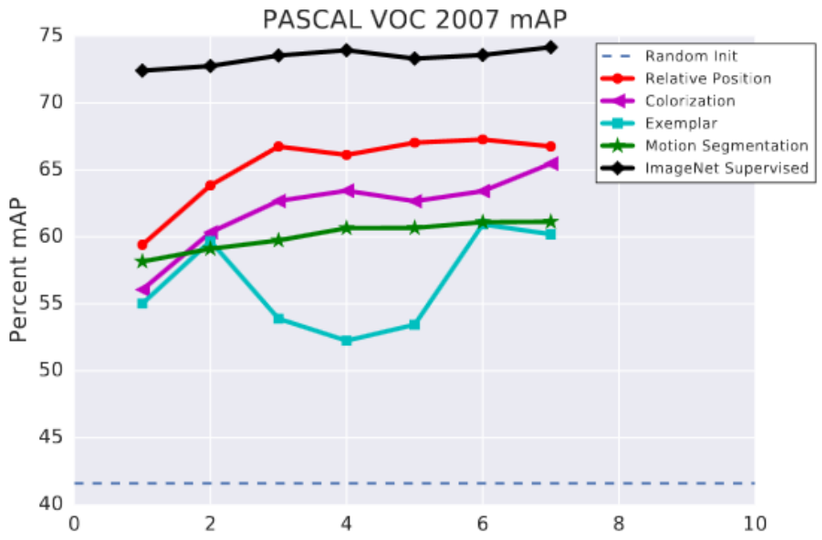
Multi-task Self-Supervised Visual Learning: эксперименты

Image classification on ImageNet
Object detection on PASCAL VOC 2007
Depth prediction on NYU V2

самообучаем сеть
замораживаем веса
линейный слой для классификации или интегрированный для двух других задач

Multi-task Self-Supervised Visual Learning: эксперименты





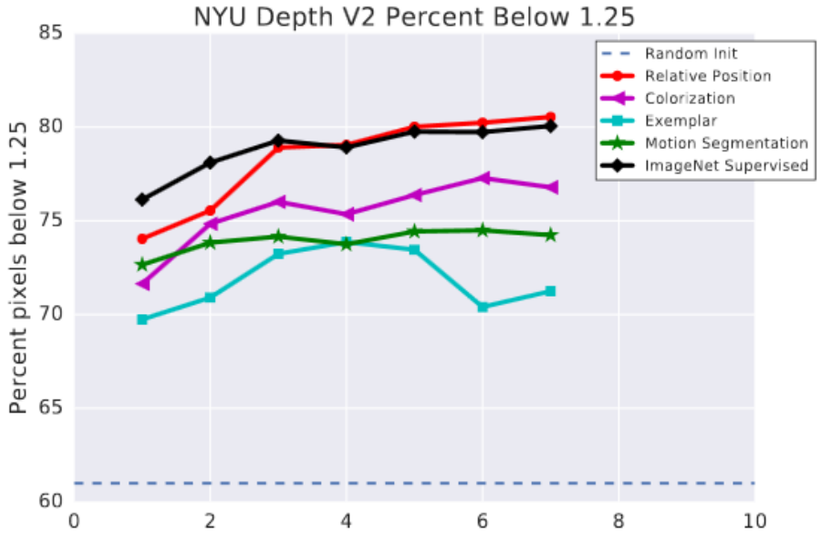
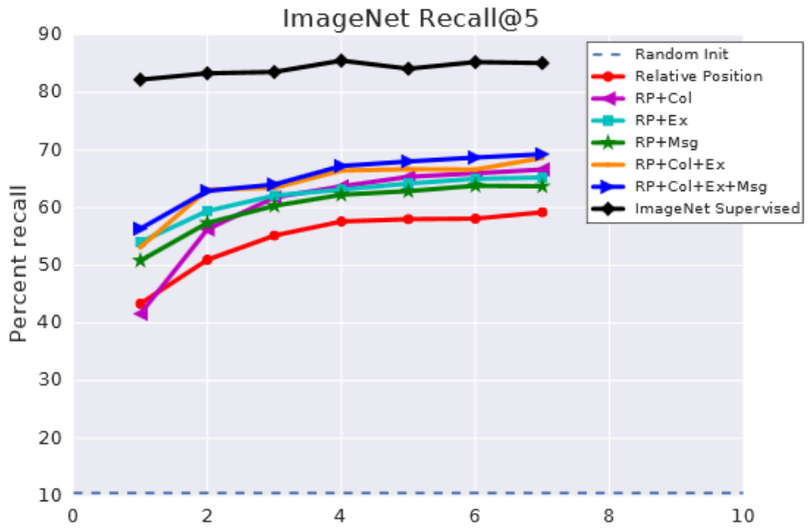
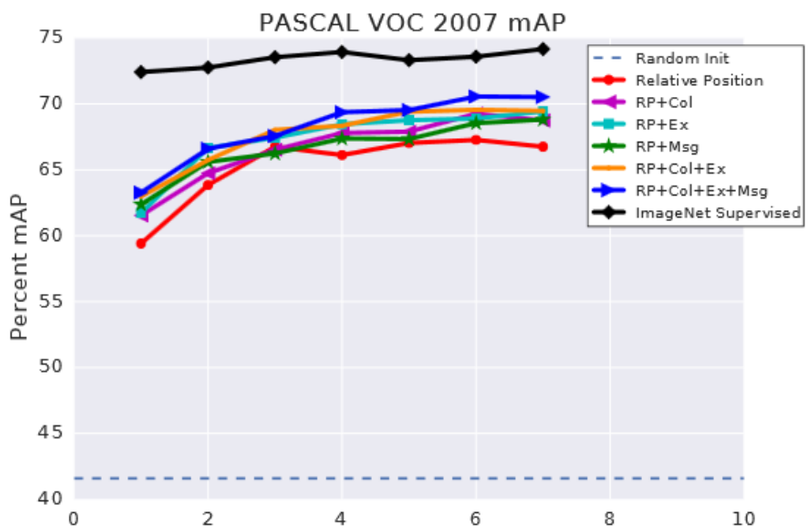


Figure 3. Comparison of performance for different self-supervised methods over time. X-axis is compute time on the self-supervised task (∼2.4K GPU hours per tick). "Random Init" shows performance with no pre-training.

Multi-task Self-Supervised Visual Learning: эксперименты





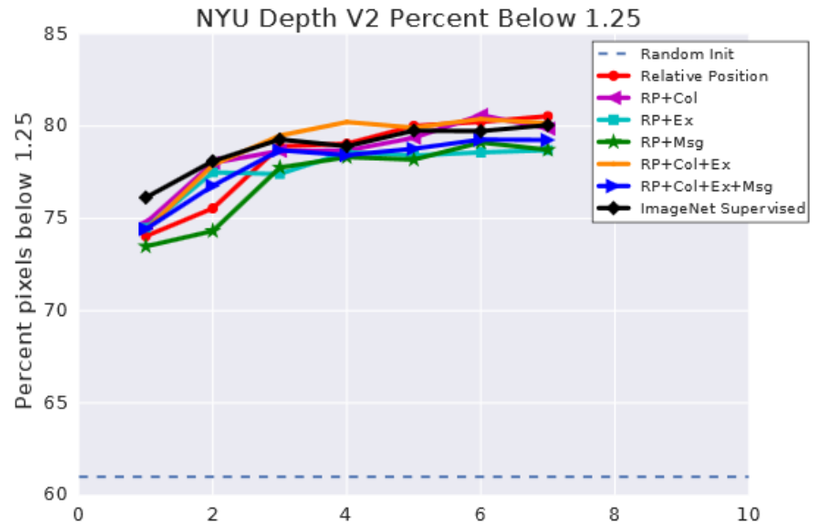


Figure 4. Comparison of performance for different multi-task self-supervised methods over time. X-axis is compute time on the self-supervised task (∼2.4K GPU hours per tick). "Random Init" shows performance with no pre-training.

Multi-task Self-Supervised Visual Learning: эксперименты

Pre-training	ImageNet	PASCAL	NYU
RP	59.21	66.75	**80.54**
RP+Col	66.64	68.75	79.87
RP+Ex	65.24	69.44	78.70
RP+MS	63.73	68.81	78.72
RP+Col+Ex	68.65	69.48	80.17
RP+Col+Ex+MS	**69.30**	**70.53**	79.25
INet Labels	85.10	74.17	80.06

Table 2. Comparison of various combinations of self-supervised tasks. Checkpoints were taken after 16.8K GPU hours, equivalent to checkpoint 7 in Figure 3. Abbreviation key: RP: Relative Position; Col: Colorization; Ex: Exemplar Nets; MS: Motion Segmentation. Metrics: ImageNet: Recall@5; PASCAL: mAP; NYU: % Pixels below 1.25.

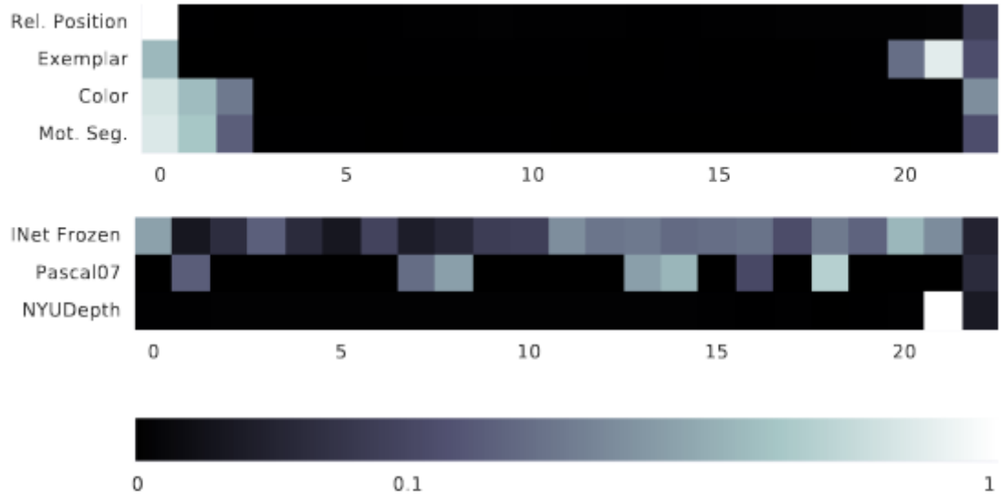


Figure 5. Weights learned via the lasso technique. Each row shows one task: self-supervised tasks on top, evaluation tasks on bottom. Each square shows $|\alpha|$ for one ResNet "Unit" (shallowest layers at the left). Whiter colors indicate higher $|\alpha|$, with a nonlinear scale to make smaller nonzero values easily visible.

Temporal coherence of color

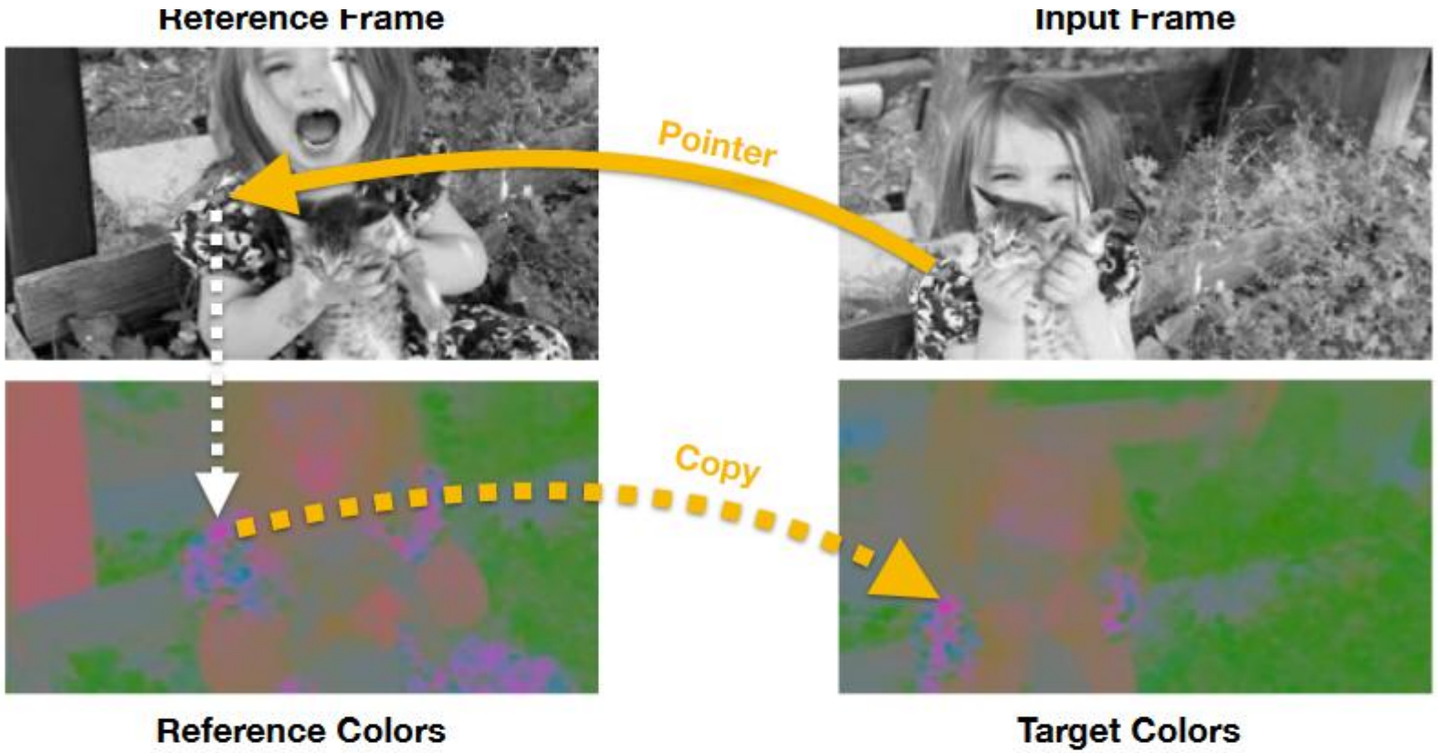


Fig. 1. Self-supervised Tracking: We capitalize on large amounts of unlabeled video to learn a self-supervised model for tracking. The model learns to predict the target colors for a gray-scale input frame by pointing to a colorful reference frame, and copying the color channels. Although we train without ground-truth labels, experiments and visualizations suggest that tracking emerges automatically in this model.

Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, Kevin Murphy

«Tracking Emerges by Colorizing Videos» https://arxiv.org/abs/1806.09594

Temporal coherence of color

раскрасить кадры видео по одному раскрашенному референсному кадру

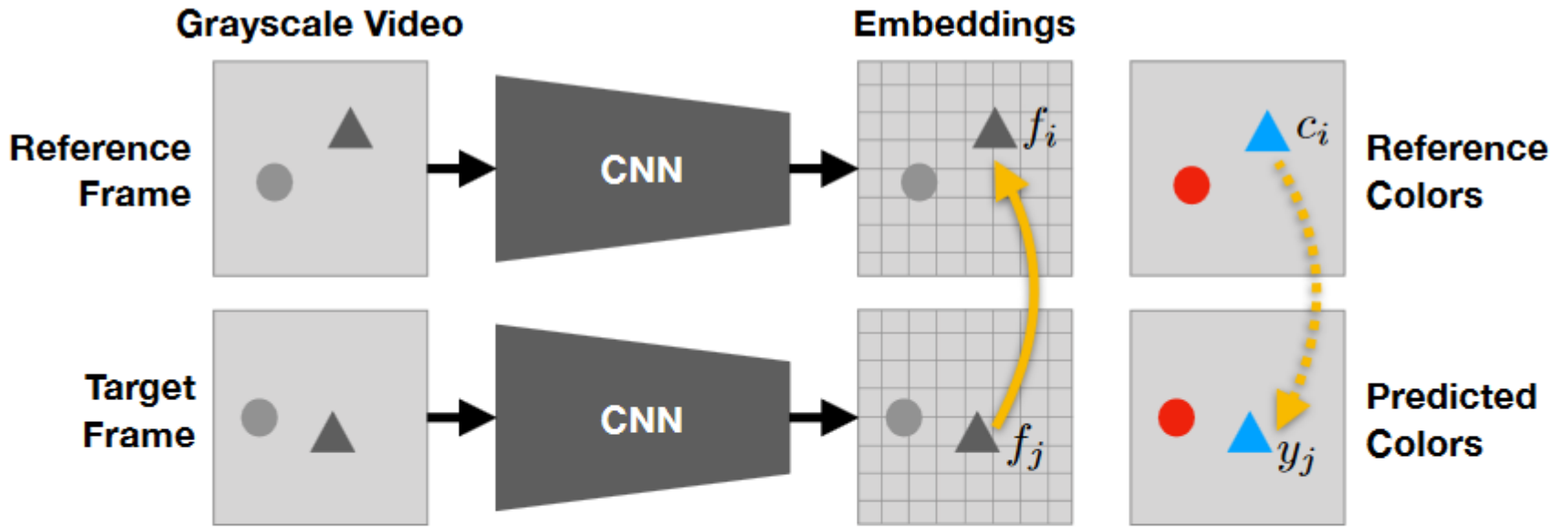
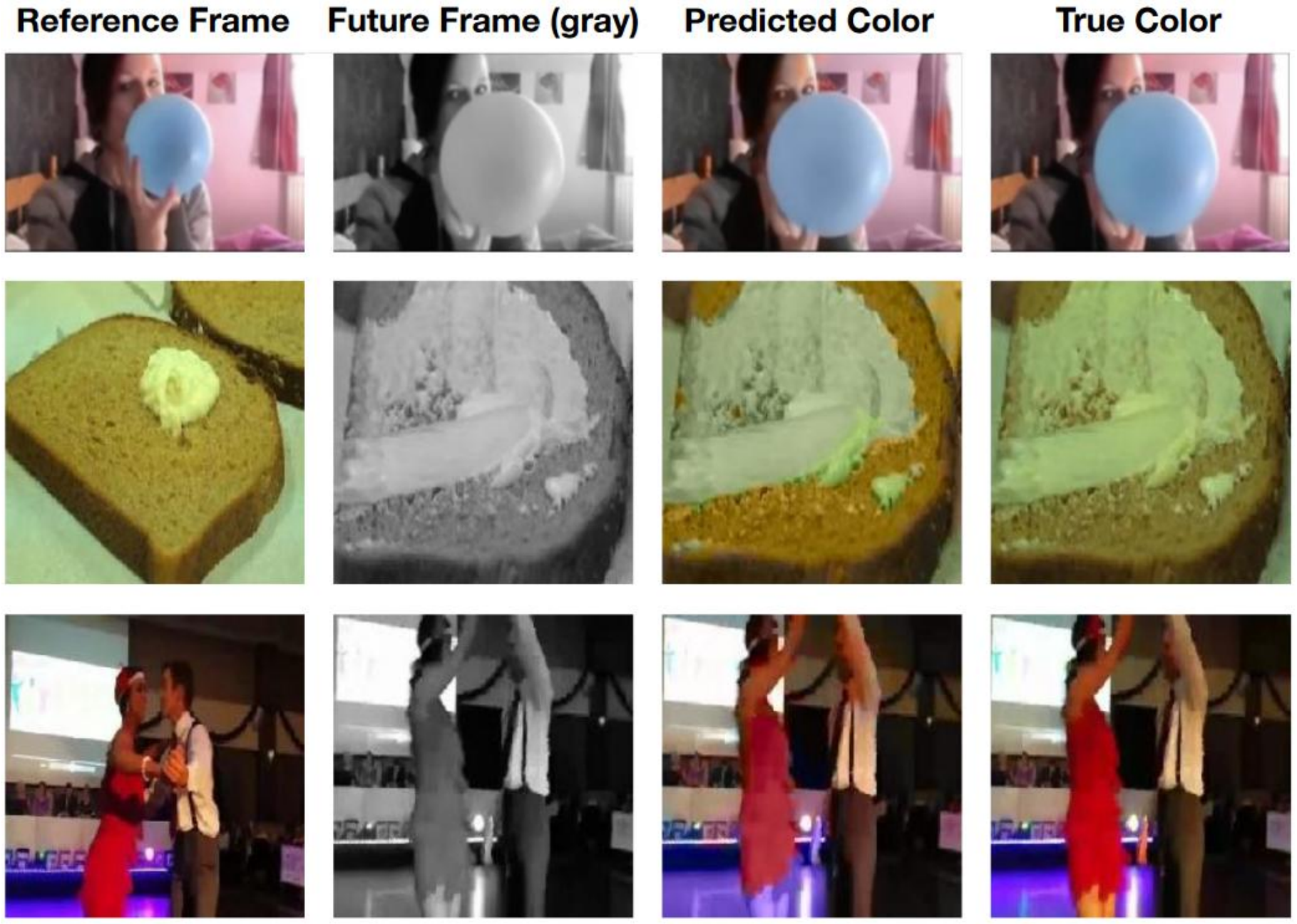


Fig. 2. Model Overview: Given gray-scale frames, the model computes low-dimensional embeddings for each location with a CNN. Using softmax similarity, the model points from the target frame into the reference frame embeddings (solid yellow arrow). The model then copies the color back into the predicted frame (dashed yellow arrow). After learning, we use the pointing mechanism as a visual tracker. Note that the model's pointer is soft, but for illustrations purposes we draw it as a single arrow.


Reference Frame
Future Frame (gray)
Predicted Color
True Color

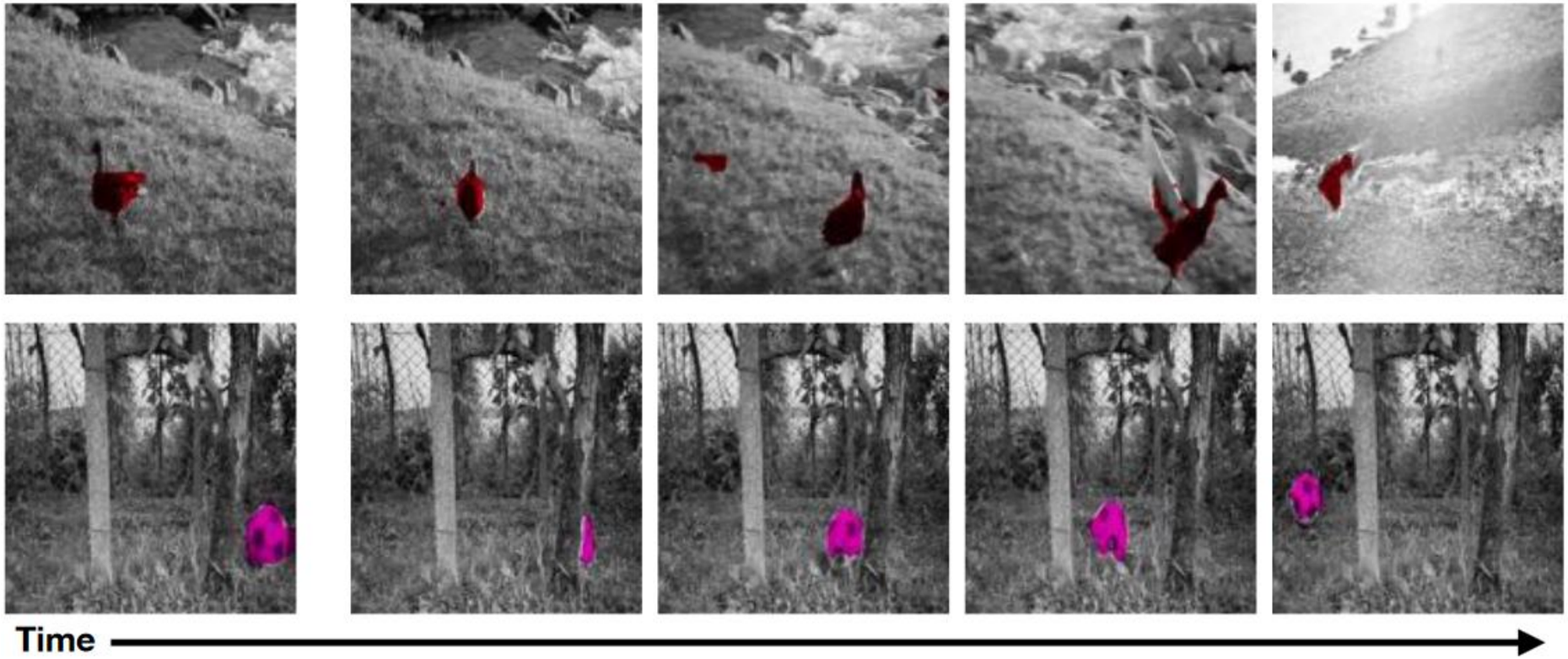


Fig. 7. Example Video Segmentations: We show results from our self-supervised model on the task of video segmentation. Colors indicate different instances. Although the model is trained without ground truth labels, the model can still propagate segmentations throughout videos. The left column shows the input frame and input masks to the model, and the rest show the predictions. Results suggest that the model is generally robust to intra-class variations, such as deformations, and occlusions. The model often handles multiple objects and cluttered backgrounds. Best viewed in color. We provide videos of results online at https://goo.gl/qjHyPK

Temporal Context-based Learning

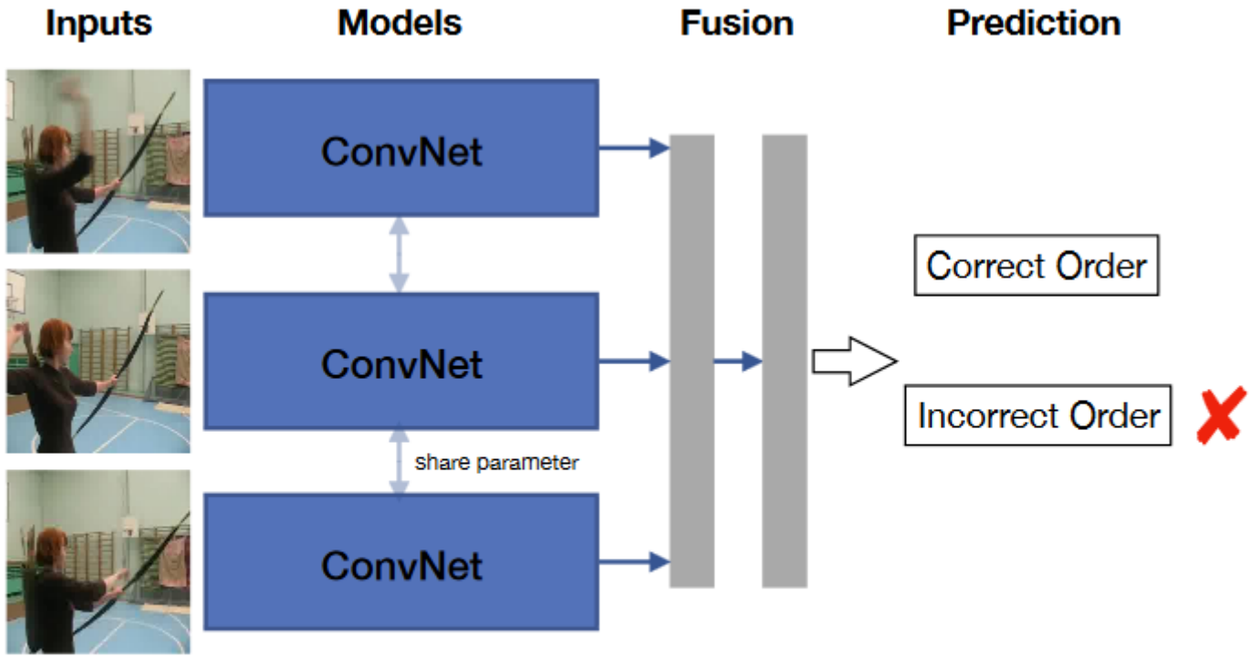


Fig. 22. The pipeline of Shuffle and Learn [40]. The network is trained to verify whether the input frames are in correct temporal order. Figure is reproduced based on [40].

Ishan Misra, C. Lawrence Zitnick, Martial Hebert «Shuffle and Learn: Unsupervised Learning usingTemporal Order Verification» // https://arxiv.org/pdf/1603.08561.pdf

Learning from Visual-Audio Correspondence

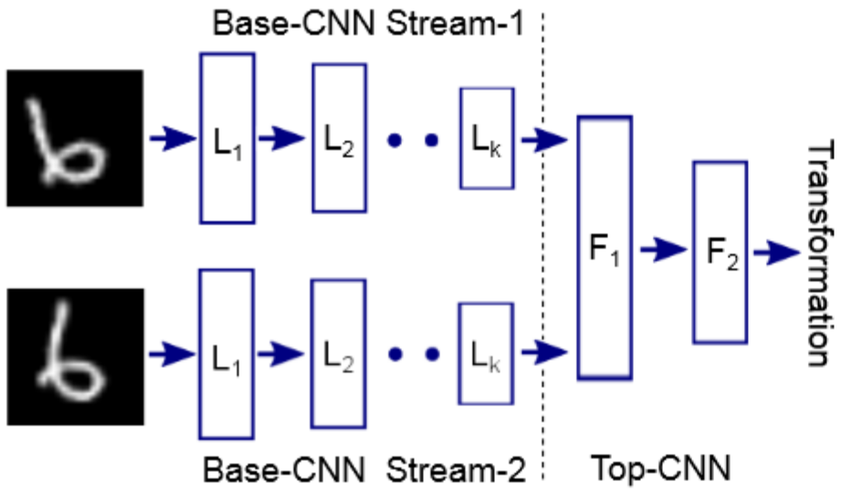


Figure 2: Description of the method for feature learning. Visual features are learnt by training a Siamese style Convolutional Neural Network (SCNN, [8]) that takes as inputs two images and predicts the transformation between the images (i.e. egomotion). Each stream of the SCNN (called as Base-CNN or BCNN) computes features for one image. The outputs of two BCNNs are concatenated and passed as inputs to a second multilayer CNN called as the Top-CNN (TCNN) (shown as layers F_1, F_2). The two BCNNs have the same architecture and share weights. After feature learning, TCNN is discarded and a single BCNN stream is used as a standard CNN for extracting features for performing target tasks like scene recognition.

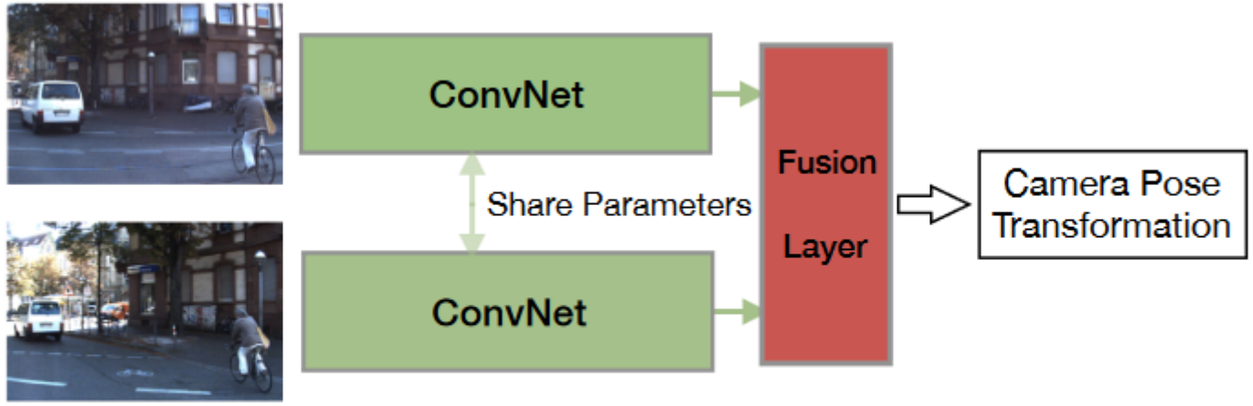


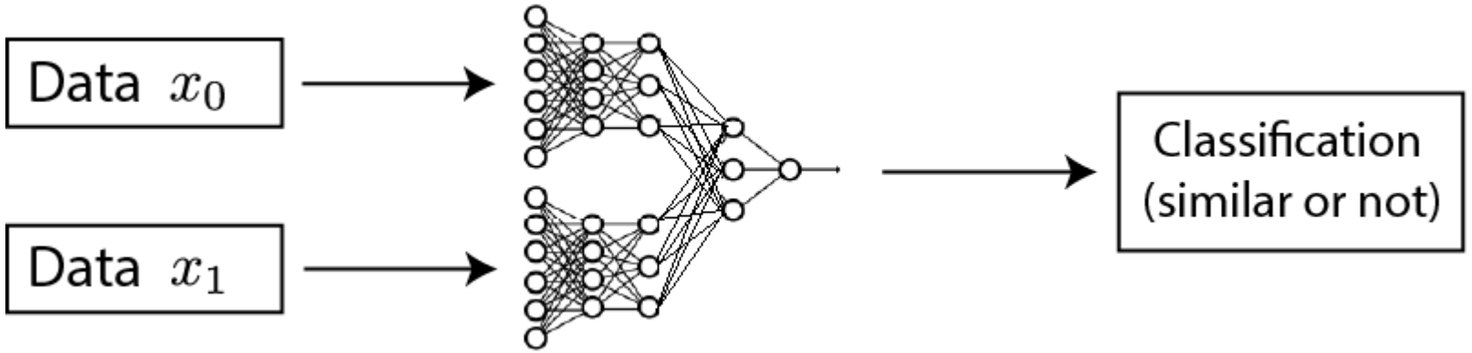
Fig. 24. The architecture of camera pose transformation estimation from egocentric videos [94].

https://arxiv.org/pdf/1505.01596.pdf

Современный подход: сравнительное обучение

Generative / Predictive

Colorization
Auto-Encoders
…



Contrastive

TCN
CPC
Deep-InfoMax
…

https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html

Современный подход: сравнительное обучение

для точки *x*, кодировщика *f*, схожести score хотим

$$\text{score}(x, x^+) >> \text{score}(x, x^-)$$

пытаемся для одного позитивного и N-1 негативного оптимизировать

$$-\mathbf{E}_X \left[\log \frac{\exp(f(x)^{\mathrm{T}} f(x^+))}{\exp(f(x)^{\mathrm{T}} f(x^+)) + \sum_{j=1}^{N-1} \exp(f(x)^{\mathrm{T}} f(x_j))} \right]$$

здесь считаем

$$\text{score}(x, x^+) = f(x)^{\mathrm{T}} f(x^+)$$

часто добавляют аугментации: *x* → *T(x)*

названия такой ошибки: InfoNCE loss / multi-class n-pair loss / ranking-based NCE

https://arxiv.org/abs/1905.06922

– доказывается, что она даёт нижнюю оценку на взаимную информацию между f(x), f(x$^+$)

Сравнительное обучение для решения задача обучения с учителем

Loss	Architecture	Top-1	Top-5
Cross Entropy (baselines)	AlexNet [27]	56.5	84.6
	VGG-19+BN [42]	74.5	92.0
	ResNet-18 [20]	72.1	90.6
	MixUp ResNet-50 [56]	77.4	93.6
	CutMix ResNet-50 [55]	78.6	94.1
	Fast AA ResNet-50 [9]	77.6	95.3
	Fast AA ResNet-200 [9]	80.6	95.3
Cross Entropy (our implementation)	ResNet-50	77.0	92.9
	ResNet-200	78.0	93.3
Supervised Contrastive	ResNet-50	**78.8**	**93.9**
	ResNet-200	**80.8**	**95.6**

Table 1: Top-1/Top-5 accuracy results on ImageNet on ResNet-50 and ResNet-200 with AutoAugment [9] being used as the augmentation for Supervised Contrastive learning. Achieving 78.8% on ResNet-50, we outperform all of the top methods whose performance is shown above. Baseline numbers are taken from the referenced papers and we also additionally re-implement cross-entropy ourselves for fair comparison.

Loss	Architecture	rel. mCE	mCE
Cross Entropy (baselines)	AlexNet [27]	100.0	100.0
	VGG-19+BN [42]	122.9	81.6
	ResNet-18 [20]	103.9	84.7
Cross Entropy (our implementation)	ResNet-50	103.7	68.4
	ResNet-200	96.6	69.4
Supervised Contrastive	ResNet-50	**87.5**	**64.4**
	ResNet-200	**77.1**	**57.2**

Table 2: Training with Supervised Contrastive Loss makes models more robust to corruptions in images, as measured by Mean Corruption Error (mCE) and relative mCE over the ImageNet-C dataset [22] (lower is better).

в задаче обучения с учителем сначала учат двойную сеть с InfoNCE loss,

а потом доучивают на кросс-энтропию по исходным меткам

Prannay Khosla et al «Supervised Contrastive Learning» // https://arxiv.org/pdf/2004.11362.pdf

Invariant Information Clustering (IIC)

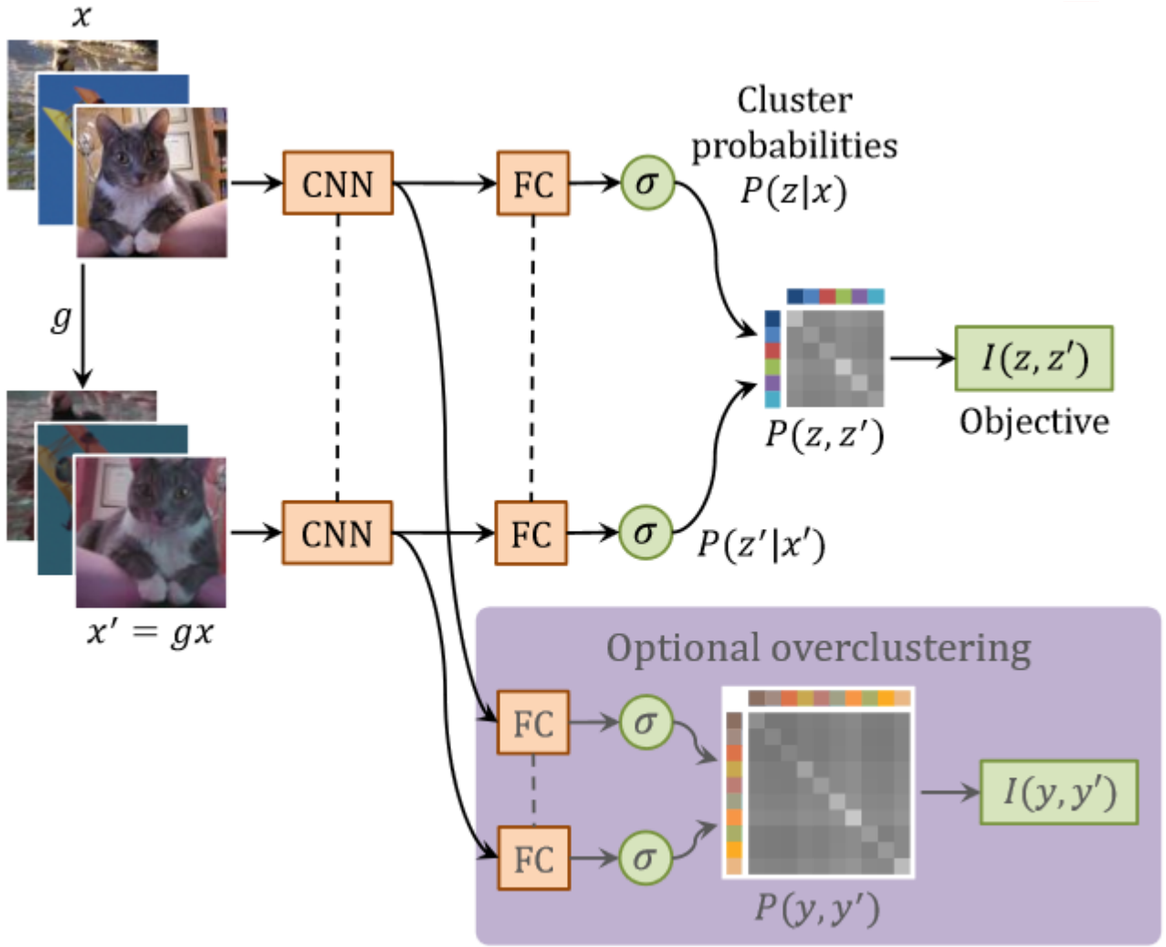


Figure 2: IIC for image clustering. Dashed line denotes shared parameters, g is a random transformation, and I denotes mutual information (eq. (3)).

X. Ji, J. F. Henriques, and A. Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In Proceedings of the IEEE International Conference onComputer Vision, pages 9865–9874, 2019, https://arxiv.org/pdf/1807.06653.pdf

Invariant Information Clustering (IIC)

Максимизация MI между разными аугментациями изображения (между выходами – распределениями)

Сеть осуществляет мягкую (soft) кластеризацию

Sobel filtering

Auxiliary overclustering head – если известны какие-то метки, то оптимизируем MI по представителям меток, а если неизвестны – AOH (по представителям объектов)

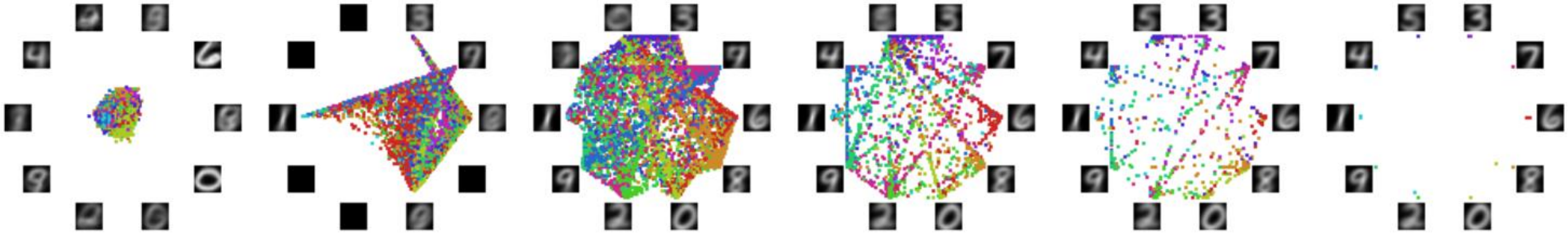


Figure 3: Training with IIC on unlabelled MNIST in successive epochs from random initialisation (left). The network directly outputs cluster assignment probabilities for input images, and each is rendered as a coordinate by convex combination of 10 cluster vertices. There is no cherry-picking as the 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).

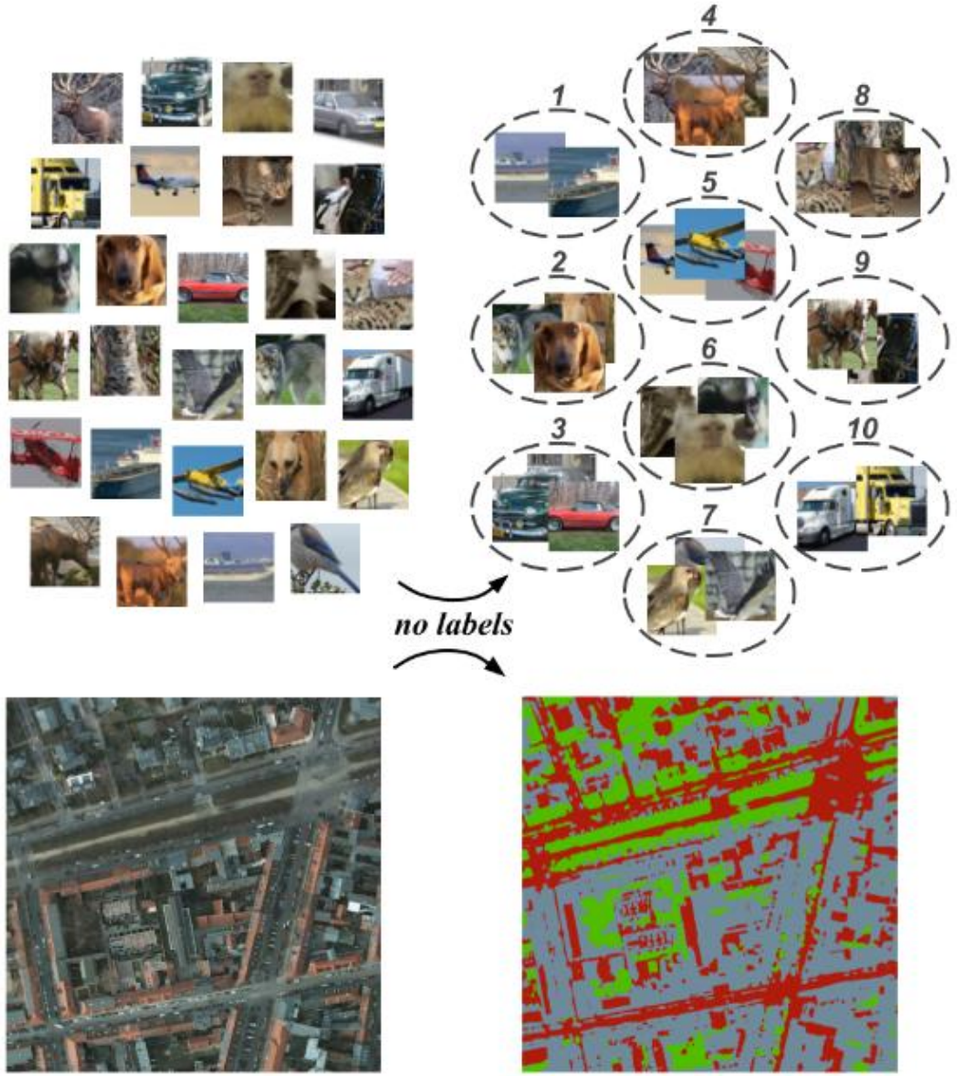


Figure 1: Models trained with IIC on entirely unlabelled data learn to cluster images (top, STL10) and patches (bottom, Potsdam-3). The raw clusters found directly correspond to semantic classes (dogs, cats, trucks, roads, vegetation etc.) with state-of-the-art accuracy. Training is end-to-end and randomly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

Deep InfoMax (DIM)

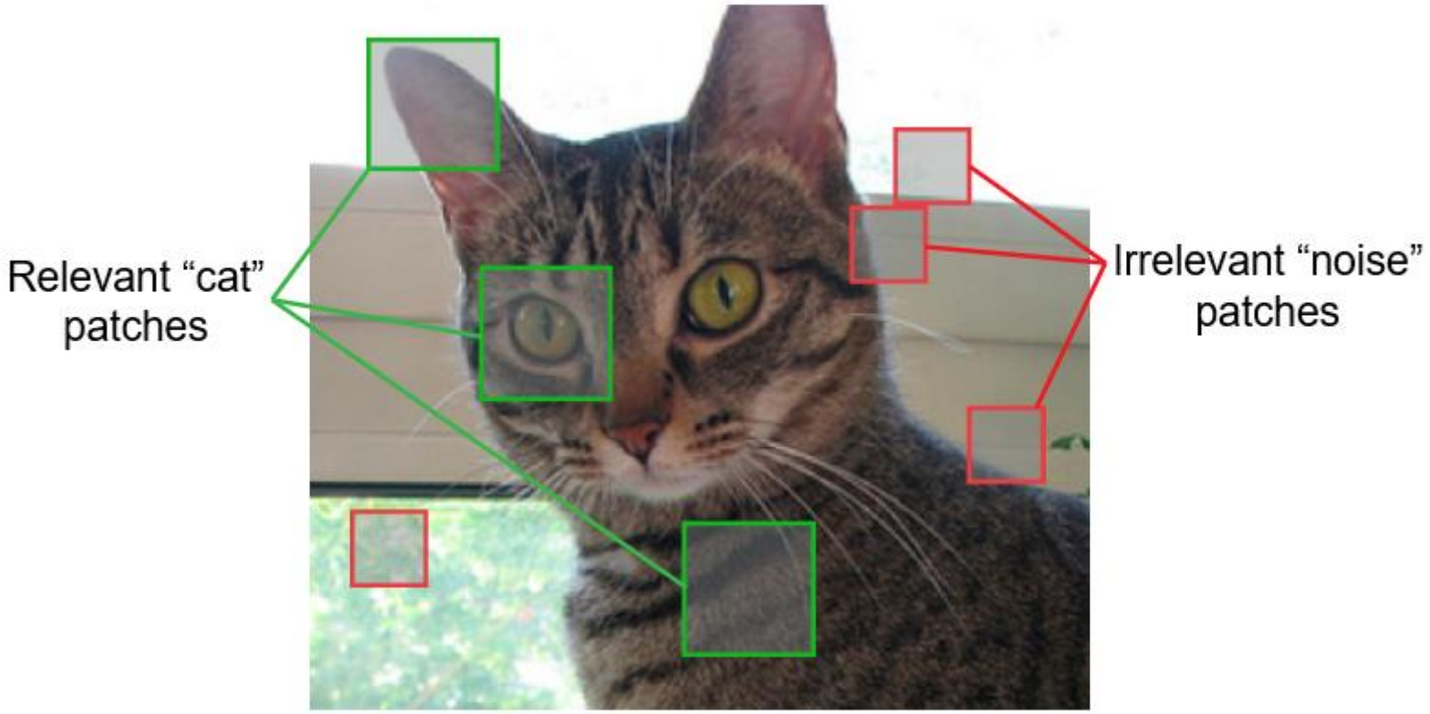
global MI – между входом и выходом кодировщика
показано, что надо смотреть на MI между патчами и выходом кодировщика

Deep InfoMax (DIM) оценивает и максимизирует MI (на самом деле InfoNCE) между входом и выучиваемым высоко-уровневым представлением

используется adversarial learning чтобы представление удовлетворяло желаемым статистическим характеристикам

R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon,K. Grewal, P. Bachman, A. Trischler, and Y. Bengio. Learning deep representations by mutual information estimation and maximization. In International Conference on Learning Representations, 2019, https://arxiv.org/abs/1808.06670

Deep InfoMax (DIM)



нельзя просто максимизировать MI – много нерелевантных фрагментов

Deep InfoMax (DIM)

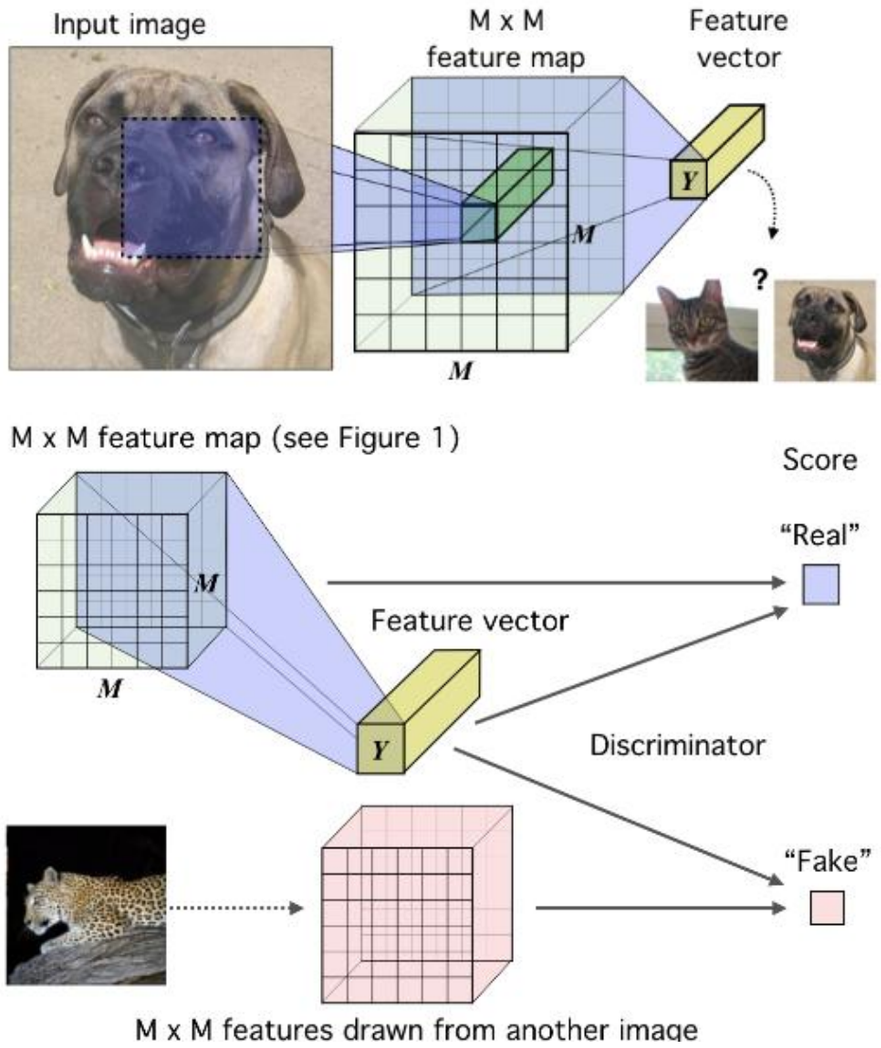


Figure 1: **The base encoder model in the context of image data.** An image (in this case) is encoded using a convnet until reaching a feature map of $M \times M$ feature vectors corresponding to $M \times M$ input patches. These vectors are summarized into a single feature vector, Y. Our goal is to train this network such that useful information about the input is easily extracted from the high-level features.

Figure 2: **Deep InfoMax (DIM) with a global MI$(X;Y)$ objective.** Here, we pass both the high-level feature vector, Y, and the lower-level $M \times M$ feature map (see Figure 1) through a discriminator to get the score. Fake samples are drawn by combining the same feature vector with a $M \times M$ feature map from another image.

глобальный вектор конкатенируется к локальным (и с чужого изображения для получения негативных примеров), оптимизируем MI(вход, рез-т конкатенации)

Deep InfoMax (DIM)

в итоге пришли к такому решению:
дискриминатор на конкатенации признака одного локального участка и глобального
(чтобы информация об изображениях хранилась и в патчах)

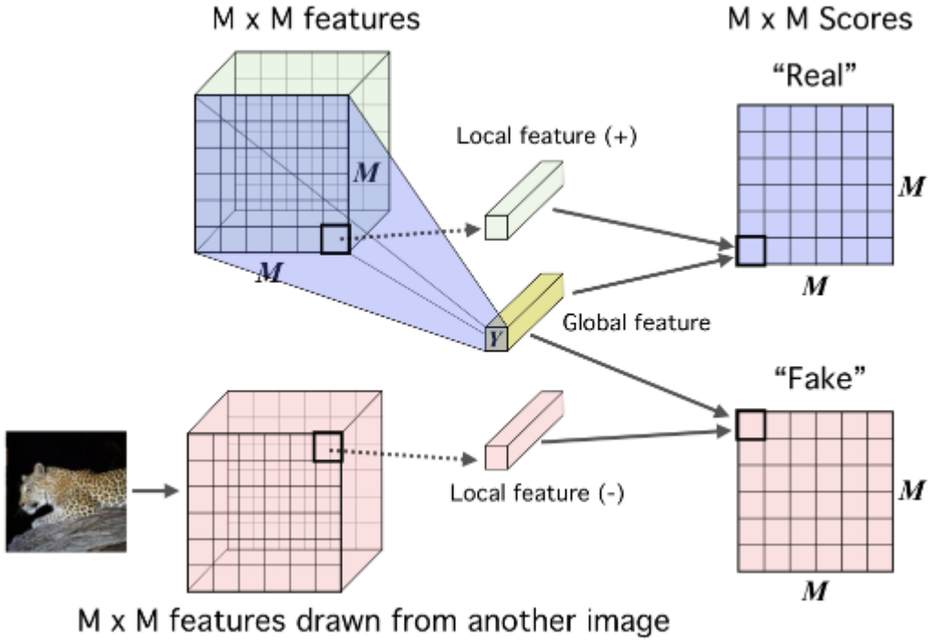


Figure 3: **Maximizing mutual information between local features and global features.** First we encode the image to a feature map that reflects some structural aspect of the data, e.g. spatial locality, and we further summarize this feature map into a global feature vector (see Figure 1). We then concatenate this feature vector with the lower-level feature map *at every location*. A score is produced for each local-global pair through an additional function (see the Appendix A.2 for details).

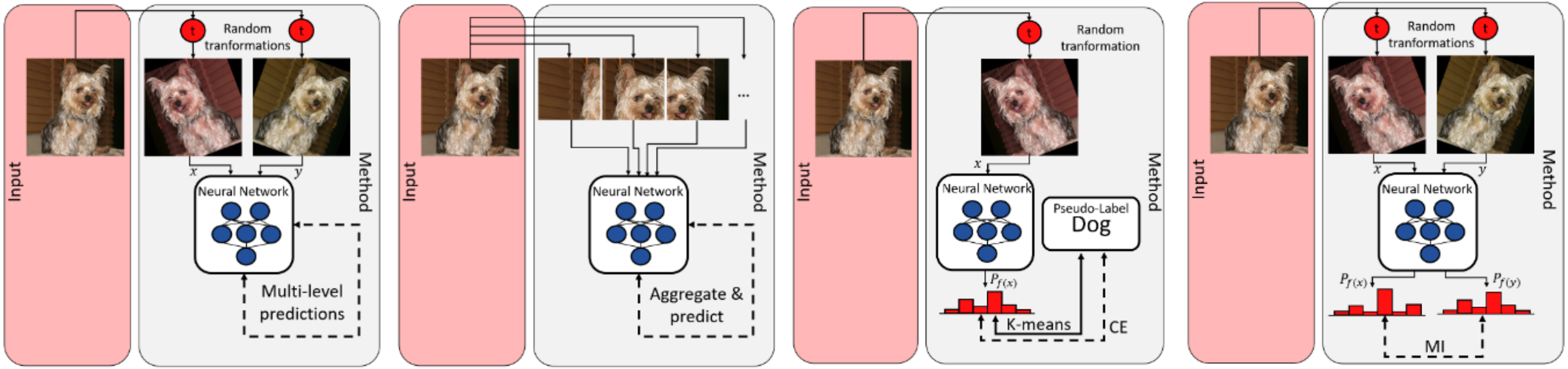
якорь – глобальный вектор
позитивный – локальный с этого изображения
негативный – локальный с другого

Deep InfoMax (DIM)

Table 1: Classification accuracy (top 1) results on CIFAR10 and CIFAR100. DIM(L) (i.e., with the local-only objective) outperforms all other unsupervised methods presented by a wide margin. In addition, DIM(L) approaches or even surpasses a fully-supervised classifier with similar architecture. DIM with the global-only objective is competitive with some models across tasks, but falls short when compared to generative models and DIM(L) on CIFAR100. Fully-supervised classification results are provided for comparison.

Model	CIFAR10			CIFAR100		
	conv	fc (1024)	Y (64)	conv	fc (1024)	Y (64)
Fully supervised		75.39			42.27	
VAE	60.71	60.54	54.61	37.21	34.05	24.22
AE	62.19	55.78	54.47	31.50	23.89	27.44
β-VAE	62.4	57.89	55.43	32.28	26.89	28.96
AAE	59.44	57.19	52.81	36.22	33.38	23.25
BiGAN	62.57	62.74	52.54	37.59	33.34	21.49
NAT	56.19	51.29	31.16	29.18	24.57	9.72
DIM(G)	52.2	52.84	43.17	27.68	24.35	19.98
DIM(L) (DV)	**72.66**	**70.60**	**64.71**	**48.52**	**44.44**	**39.27**
DIM(L) (JSD)	**73.25**	**73.62**	**66.96**	**48.13**	**45.92**	**39.60**
DIM(L) (infoNCE)	**75.21**	**75.57**	**69.13**	**49.74**	**47.72**	**41.61**

Augmented Multiscale Deep InfoMax (AMDIM)



(a) AMDIM (b) CPC (c) DeepCluster (d) IIC

Figure 6: Illustration of four selected self-supervised methods - The used method is given below each image. The input is given in the red box on the left side. On the right side an illustration of the method is provided. The fine-tuning part is excluded. In general the process is organized from top to bottom. At first the input images are either preprocessed by one or two random transformations or are split up. The following neural network uses these preprocessed images (x, y) as input. The calculation of the loss (dotted line) is different for each method. AMDIM and CPC use internal elements of the network to calculate the loss. DeepCluster and IIC use the predicted output distribution $(P_{f(x)}, P_{f(y)})$ to calculate a loss. For further details see the corresponding entry in section 3.

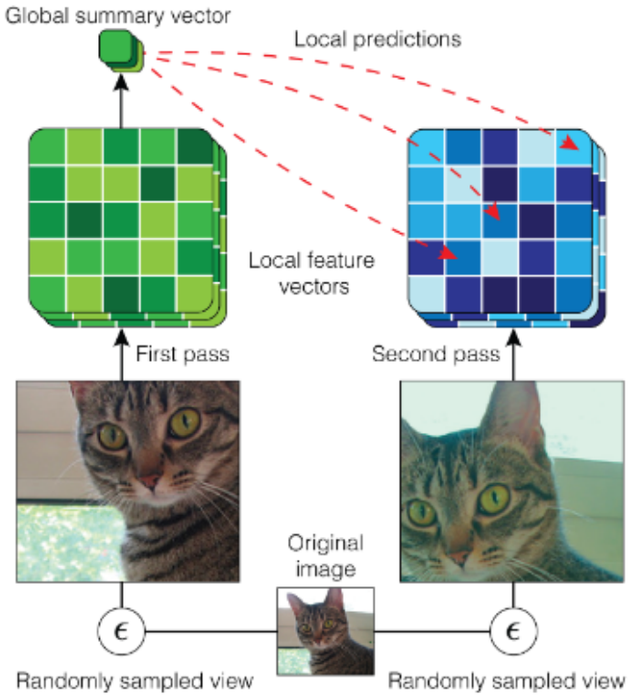
Augmented Multiscale Deep InfoMax (AMDIM)

Расширяем DIM:

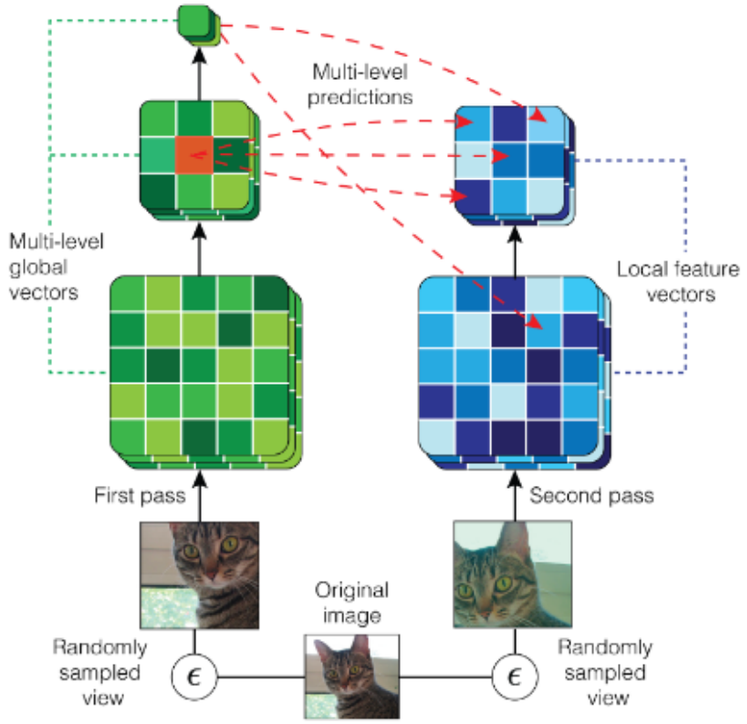
1) максимизация MI между локальными регионами разных аугментаций

2) между разными уровнями

P. Bachman, R. D. Hjelm, and W. Buchwalter. Learning representations by maximizing mutual information across views. In Advances in Neural Information Processing Systems, pages 15509–15519, 2019. https://arxiv.org/abs/1906.00910



(a)



(b)

Algorithm Compute and Memory-Efficient NCE

// n_a: # antecedents, n_c: # consequents/antecedent
// s: array of $\phi(f_a)^\top \phi(f_c)$ scores, with size (n_a, n_a, n_c)
// the tuple after each statement gives result size

$s_{shift} = \max(\max(s, \text{dim}=2), \text{dim}=1)$ // $(n_a, 1, 1)$
$s_{exp} = \exp(s - s_{shift})$ // (n_a, n_a, n_c)
$s_{self} = \text{sum}(s_{exp}, \text{dim}=2)$ // $(n_a, n_a, 1)$
$s_{full} = \text{sum}(s_{self}, \text{dim}=1)$ // $(n_a, 1, 1)$
$s_{other} = s_{full} - s_{self}$ // $(n_a, n_a, 1)$
$s_{lse} = \log(s_{exp} + s_{other})$ // (n_a, n_a, n_c)
$s_{nce} = s - s_{shift} - s_{lse}$ // (n_a, n_a, n_c)
$\ell_{nce} = -\frac{1}{n_a n_c} \sum_{i=1}^{n_a} \sum_{j=1}^{n_c} s_{nce}[i, i, j]$

Algorithm ImageNet Encoder Architecture

ReLU(Conv2d(3, ndf, 5, 2, 2))
ReLU(Conv2d(ndf, ndf, 3, 1, 0))
ResBlock(1*ndf, 2*ndf, 4, 2, ndepth)
ResBlock(2*ndf, 4*ndf, 4, 2, ndepth)
ResBlock(4*ndf, 8*ndf, 2, 2, ndepth) – provides f_7
ResBlock(8*ndf, 8*ndf, 3, 1, ndepth) – provides f_5
ResBlock(8*ndf, 8*ndf, 3, 1, ndepth)
ResBlock(8*ndf, nrkhs, 3, 1, 1) – provides f_1

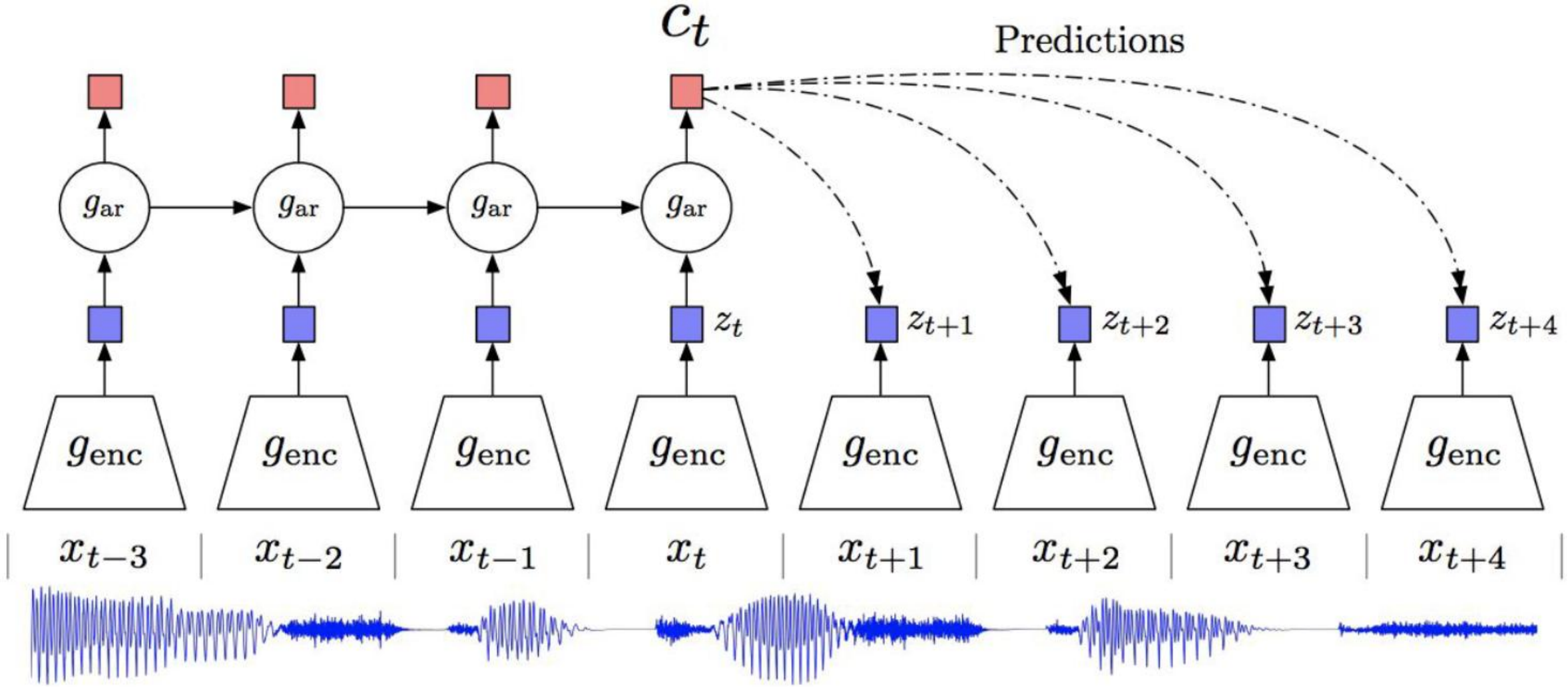
(c)

Figure 1: **(a)**: Local DIM with predictions across views generated by data augmentation. **(b)**: Augmented Multiscale DIM, with multiscale infomax across views generated by data augmentation. **(c)**-top: An algorithm for efficient NCE with minibatches of n_a images, comprising one antecedent and n_c consequents per image. For each true (antecedent, consequent) positive sample pair, we compute the NCE bound using all consequents associated with all other antecedents as negative samples. Our pseudo-code is roughly based on pytorch. We use dynamic programming in the log-softmax normalizations required by ℓ_{nce}. **(c)**-bottom: Our ImageNet encoder architecture.

	Architecture	Publication	CIFAR-10	CIFAR-100	STL-10	ILSVRC-2012
Supervised (100% labels)	Best reported	–	97.14[42]	79.82[2]	68.7 [18]	78.57 [49] / 94.10* [49]
Self-Supervised Methods						
AMDIM [2]	ResNet18 [16]	2019	91.3† / 93.6†‡	70.2† / 73.8†‡	93.6 / 93.8‡	60.2† / 60.9†‡
Context [11]	ResNet50 [16]	2015				51.4† [24]
CPC [43, 17]	ResNet-170 [17]	2019	77.45† [18]		77.81† [18]	61.0 / 84.88*
DeepCluster [4]	AlexNet [27]	2018			73.4 [21]	41†
DIM [18]	AlexNet [27]	2019			72.57‡	
DIM [18]	GAN Discriminator [39]	2019	75.21†‡	49.74†‡		
Exemplar [12]	ResNet50 [16]	2016				46.0† [24] / 81.01* [49]
IIC [21]	ResNet34 [16]	2019			88.8	
Jigsaw [35]	AlexNet [27]	2016				44.6† [24]
Rotation [14]	AlexNet [27]	2018				55.4† [24]
Rotation [14]	ResNet50v2 [16]	2018				78.53* [49]

https://arxiv.org/pdf/2002.08721.pdf

Contrastive Predictive Coding (CPC)



кодировщик в латентное представление + авторегрессионная модель

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding

https://arxiv.org/abs/1807.03748

CPC: Mutual Information

тоже есть рассуждения, что надо смотреть на взаимную информацию при прогнозе

не совсем прогнозируем, а делаем так:

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$$
$$= \exp\left(z_{t+k}^T W_k c_t\right)$$

для downstream tasks можно использовать z_t или c_t

CPC: Noise-Constrative Estimation (NCE) / Importance Sampling

~ обучение примерно как в word2vec (т.к. классов очень много)

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

в X один позитивный объект $p(x_{t+k}|c_t)$

и негативные из распределения $p(x_{t+k})$

назвали InfoNCE-loss

ниже на выученных CPC-признаках для конкретных задач обучают логистическую регрессию

CPC-Audio: эксперименты на LibriSpeech

классификация спикеров / фонем

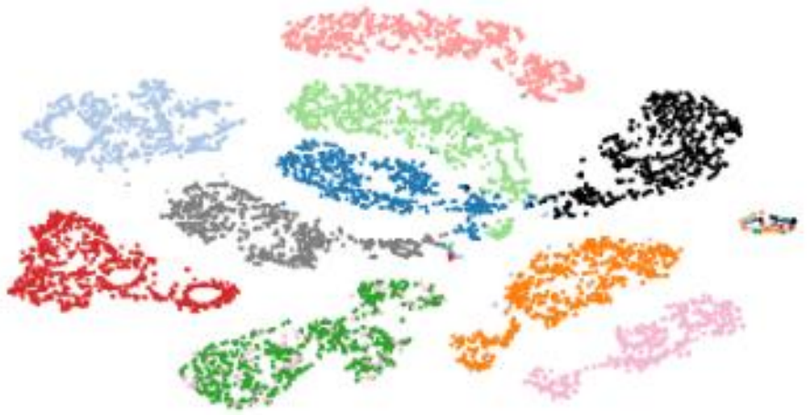


Figure 2: t-SNE visualization of audio (speech) representations for a subset of 10 speakers (out of 251). Every color represents a different speaker.

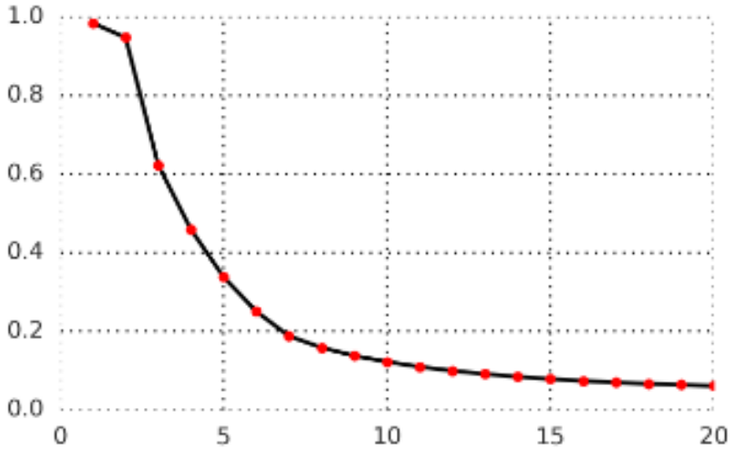


Figure 3: Average accuracy of predicting the positive sample in the contrastive loss for 1 to 20 latent steps in the future of a speech waveform. The model predicts up to 200ms in the future as every step consists of 10ms of audio.

Method	ACC
Phone classification	
Random initialization	27.6
MFCC features	39.7
CPC	64.6
Supervised	74.6
Speaker classification	
Random initialization	1.87
MFCC features	17.6
CPC	97.4
Supervised	98.5

Table 1: LibriSpeech phone and speaker classification results. For phone classification there are 41 possible classes and for speaker classification 251. All models used the same architecture and the same audio input sizes.

CPC-Image: эксперименты на ImageNet

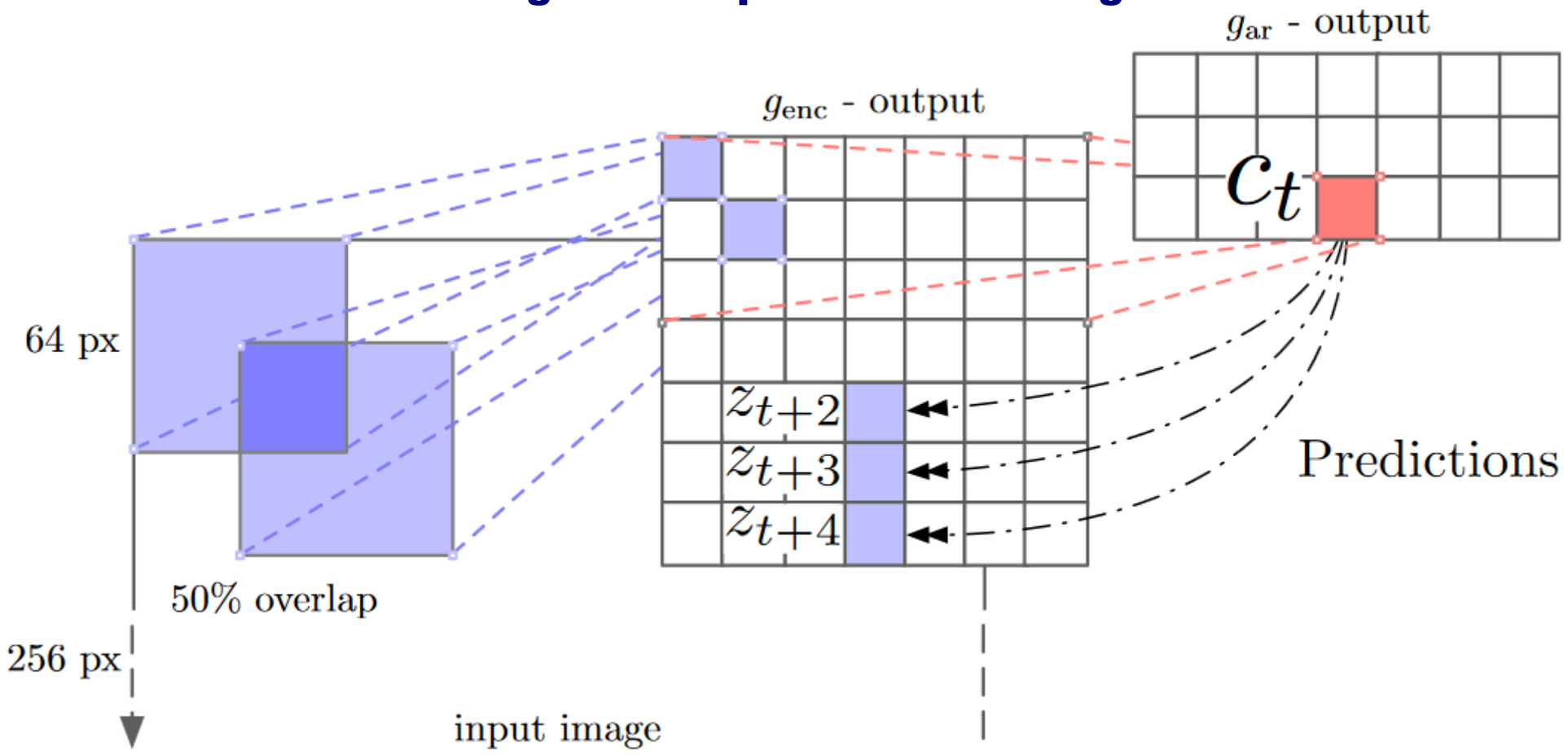


Figure 4: Visualization of Contrastive Predictive Coding for images (2D adaptation of Figure 1).

сетка 7×7 с перекрытиями

каждый кроп кодируется с помощью ResNet-v2-101 encoder (1024 признака×7×7)

PixelCNN-style autoregressive model для предсказания сверху-вниз

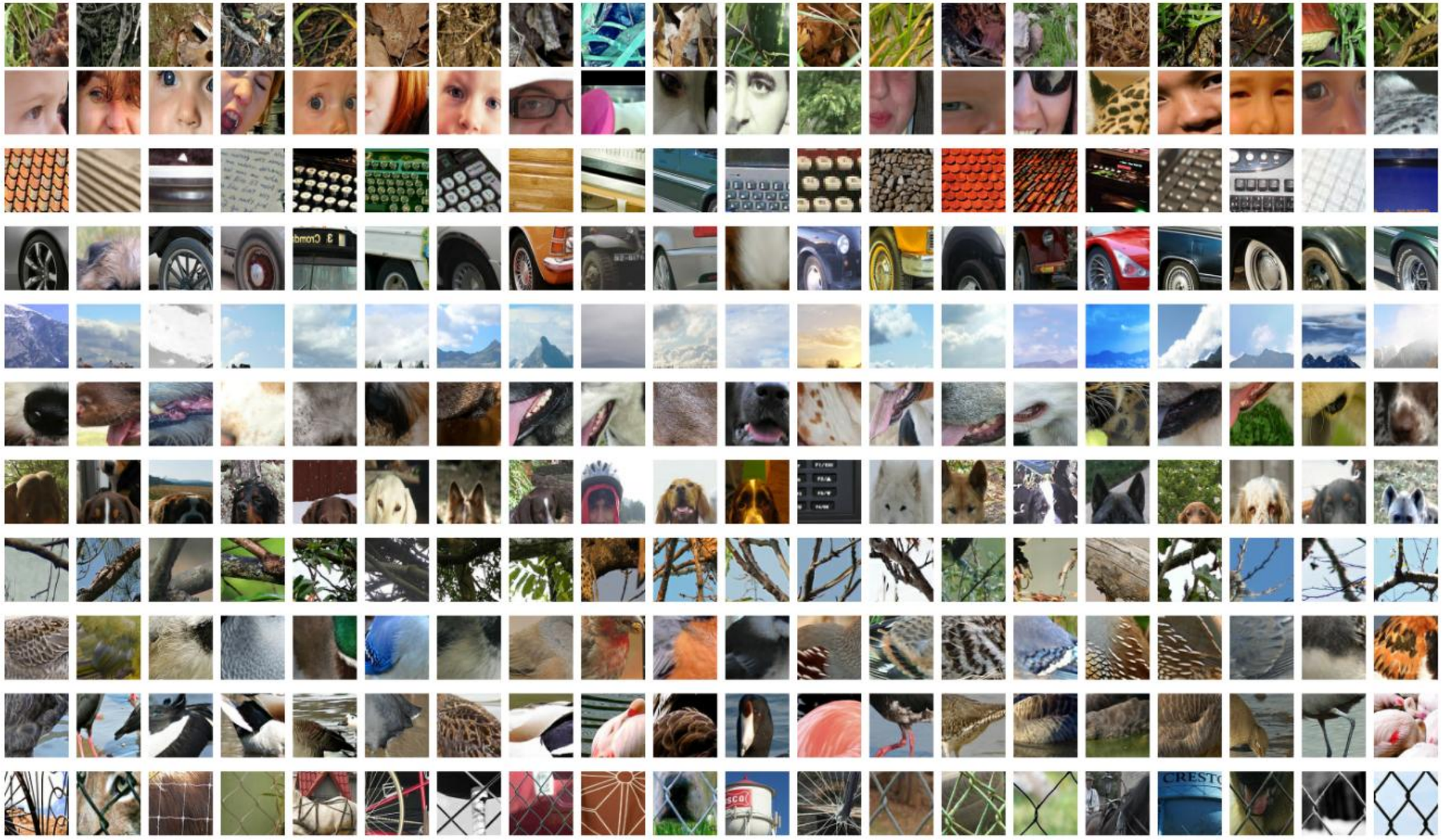


Figure 5: Every row shows image patches that activate a certain neuron in the CPC architecture.

CPC-Image: эксперименты на ImageNet

Method	Top-1 ACC
Using AlexNet conv5	
Video [28]	29.8
Relative Position [11]	30.4
BiGan [35]	34.8
Colorization [10]	35.2
Jigsaw [29] *	38.1
Using ResNet-V2	
Motion Segmentation [36]	27.6
Exemplar [36]	31.5
Relative Position [36]	36.2
Colorization [36]	39.6
CPC	**48.7**

Table 3: ImageNet top-1 unsupervised classification results. *Jigsaw is not directly comparable to the other AlexNet results because of architectural differences.

Method	Top-5 ACC
Motion Segmentation (MS)	48.3
Exemplar (Ex)	53.1
Relative Position (RP)	59.2
Colorization (Col)	62.5
Combination of	
MS + Ex + RP + Col	69.3
CPC	**73.6**

Table 4: ImageNet top-5 unsupervised classification results. Previous results with MS, Ex, RP and Col were taken from [36] and are the best reported results on this task.

CPC-NLP: эксперименты на BookCorpus

Method	MR	CR	Subj	MPQA	TREC
Paragraph-vector [40]	74.8	78.1	90.5	74.2	91.8
Skip-thought vector [26]	75.5	79.3	92.1	86.9	91.4
Skip-thought + LN [41]	79.5	82.6	93.4	89.0	-
CPC	76.9	80.1	91.2	87.7	96.8

Table 5: Classification accuracy on five common NLP benchmarks. We follow the same transfer learning setup from Skip-thought vectors [26] and use the BookCorpus dataset as source. [40] is an unsupervised approach to learning sentence-level representations. [26] is an alternative unsupervised learning approach. [41] is the same skip-thought model with layer normalization trained for 1M iterations.

movie review sentiment (MR)

customer product reviews (CR)

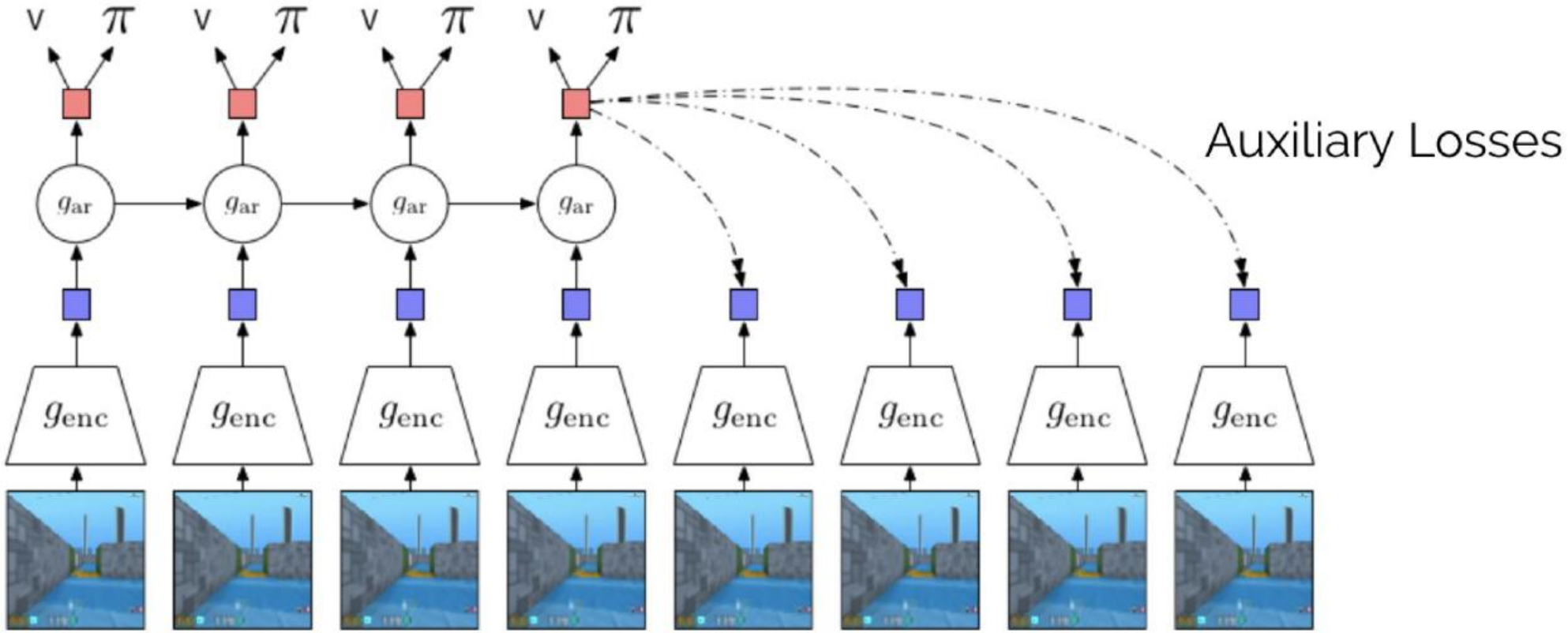
subjectivity/objectivity (Subj)

opinion polarity (MPQA)

question-type classification (TREC)

PCP-RL: эксперименты на DeepMind Lab

Auxiliary loss is on policy
Predict 30 steps in the future



PCP-RL: эксперименты на DeepMind Lab

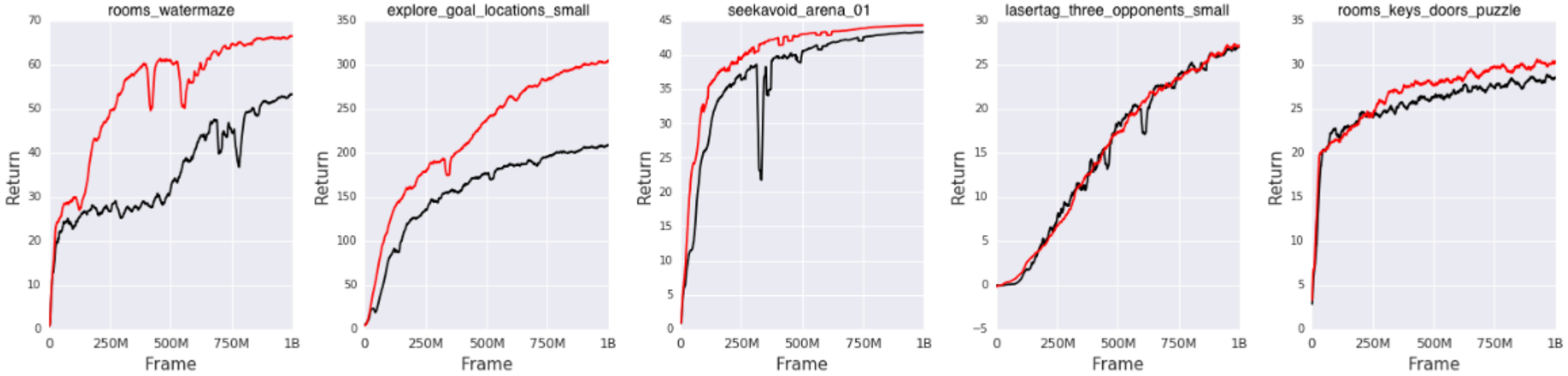


Figure 6: Reinforcement Learning results for 5 DeepMind Lab tasks used in [50]. Black: batched A2C baseline, Red: with auxiliary contrastive loss.

тут немного не так, как в других задачах

standard batched A2C agent + CPC auxiliary loss

Contrastive Predictive Coding – развитие идеи (CPC v2)

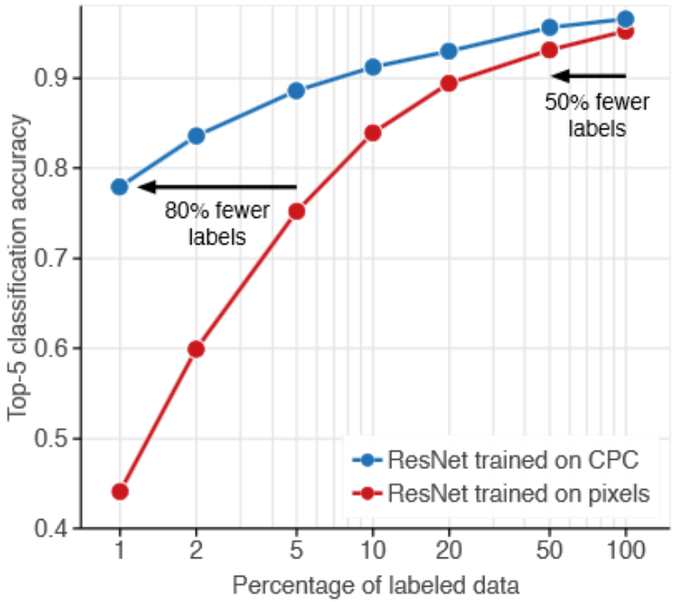


Figure 1: Data-efficient image recognition with Contrastive Predictive Coding. With decreasing amounts of labeled data, supervised networks trained on pixels fail to generalize (red). When trained on unsupervised representations learned with CPC, these networks retain a much higher accuracy in this low-data regime (blue). Equivalently, the accuracy of supervised networks can be matched with significantly fewer labels.

Olivier J. Henaff, Ali Razavi, Carl Doersch, S. M. Ali Eslami, Aaron van den Oord «Data-Efficient Image Recognition with Contrastive Predictive Coding» https://arxiv.org/abs/1905.09272

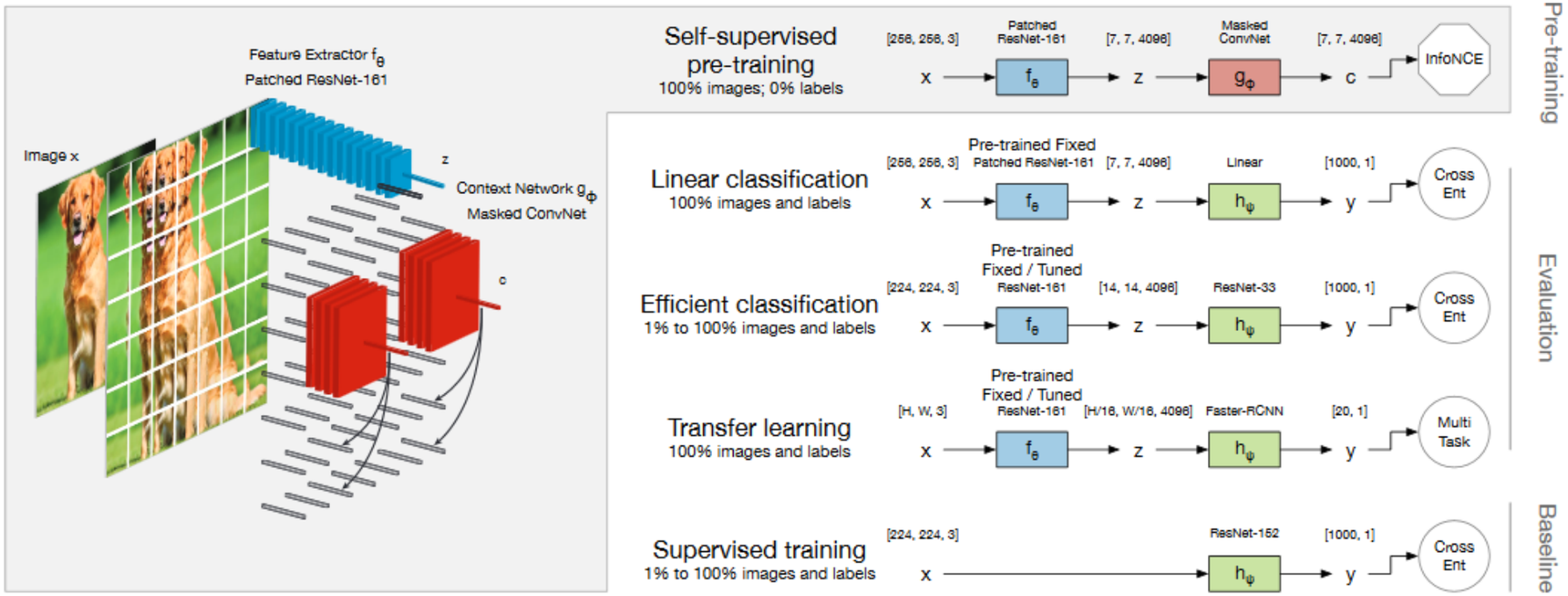


Figure 2: Overview of the framework for semi-supervised learning with Contrastive Predictive Coding. Left: unsupervised pre-training with the spatial prediction task (See Section 2.1). First, an image is divided into a grid of overlapping patches. Each patch is encoded independently from the rest with a feature extractor (blue) which terminates with a mean-pooling operation, yielding a single feature vector for that patch. Doing so for all patches yields a field of such feature vectors (wireframe vectors). Feature vectors above a certain level (in this case, the center of the image) are then aggregated with a context network (red), yielding a row of context vectors which are used to linearly predict features vectors below. Right: using the CPC representation for a classification task. Having trained the encoder network, the context network (red) is discarded and replaced by a classifier network (green) which can be trained in a supervised manner. In some experiments, we also fine-tune the encoder network (blue) for the classification task. When applying the encoder to cropped patches (as opposed to the full image) we refer to it as a *patched* ResNet in the figure.

Эволюция CPC (CPC v2)

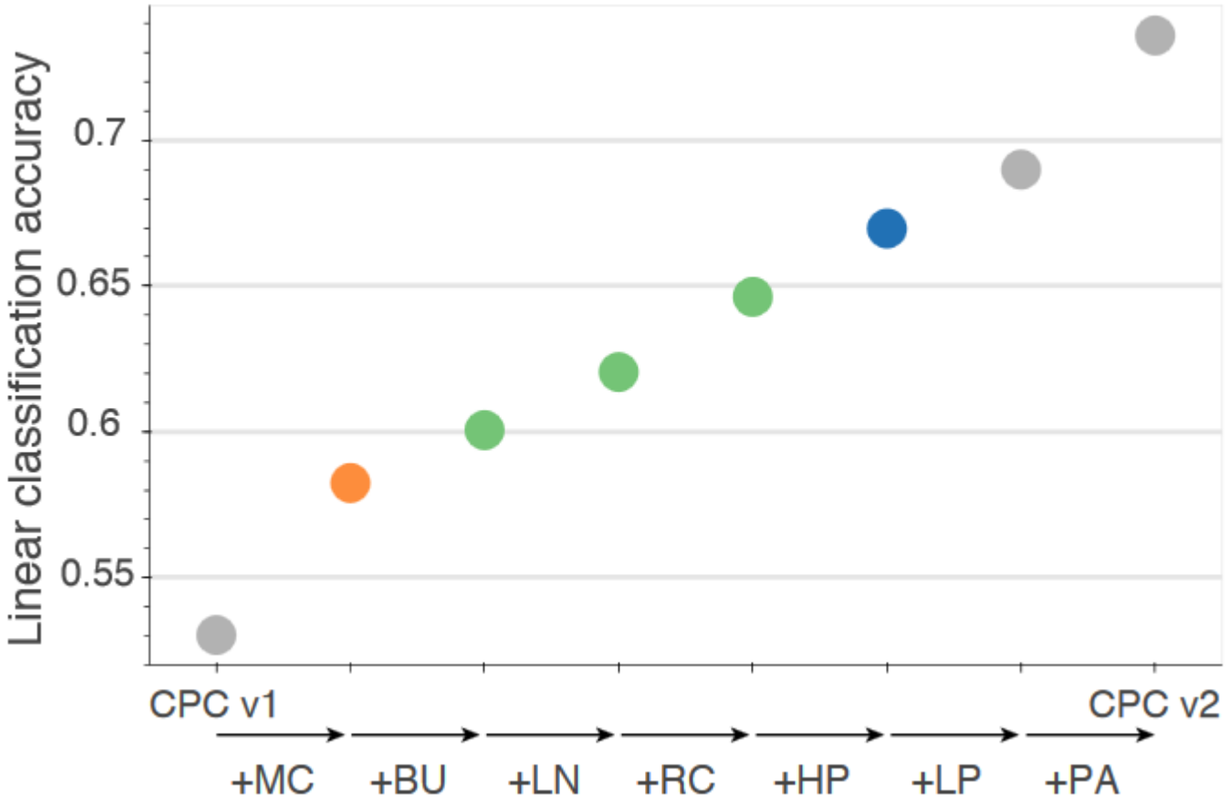


Figure 3: Linear classification performance of new variants of CPC, which incrementally add a series of modifications. MC: model capacity. BU: bottom-up spatial predictions. LN: layer normalization. RC: random color-dropping. HP: horizontal spatial predictions. LP: larger patches. PA: further patch-based augmentation. We use color to indicate the number of spatial predictions used (orange, green, blue for 1, 2 and 4 directions). Note that these accuracies are evaluated on a custom validation set and are therefore not directly comparable to the results we report and compare to.

Contrastive Predictive Coding – развитие идеи (CPC v2)

Table 1: Linear classification accuracy, and comparison to other self-supervised methods. In all cases the feature extractor is optimized in an unsupervised manner (using one of the methods listed below), and a linear classifier is trained on top using all labels in the ImageNet dataset.

Method	Architecture	Parameters (M)	Top-1	Top-5
Methods using ResNet-50:				
Local Aggregation [66]	ResNet-50	24	60.2	-
Momentum Contrast [25]	ResNet-50	24	60.6	-
CPC v2	ResNet-50	24	**63.8**	**85.3**
Methods using different architectures:				
Multi-task [13]	ResNet-101	28	-	69.3
Rotation [32]	RevNet-50 ×4	86	55.4	-
CPC v1 [58]	ResNet-101	28	48.7	73.6
BigBiGAN [15]	RevNet-50 ×4	86	61.3	81.9
AMDIM [5]	Custom-103	626	68.1	-
CMC [57]	ResNet-50 ×2	188	68.4	88.2
Momentum Contrast [25]	ResNet-50 ×4	375	68.6	-
CPC v2	ResNet-161	305	**71.5**	**90.1**

Table 2: Comparison to other methods for semi-supervised learning. *Representation learning* methods use a classifier to discriminate an unsupervised representation, and optimize it solely with respect to labeled data. *Label-propagation* methods on the other hand further constrain the classifier with smoothness and entropy criteria on unlabeled data, making the additional assumption that all training images fit into a single (unknown) testing category. *Fixed* and *fine-tuned* denote whether the feature extractor is allowed to accommodate the supervised objective. The $^{\star\#}$ markers highlight comparisons showing gains in data-efficiency relative to supervised learning.

Method	**Architecture**	**Top-5 accuracy**				
Labeled data		1%	5%	10%	50%	100%
†Supervised baseline	ResNet-200	44.1	75.2*	83.9	93.1	95.2$^{\#}$
Methods using label-propagation:						
Pseudolabeling [63]	ResNet-50	51.6	-	82.4	-	-
VAT + Entropy Minimization [63]	ResNet-50	47.0	-	83.4	-	-
Unsup. Data Augmentation [61]	ResNet-50	-	-	88.5	-	-
Rotation + VAT + Ent. Min. [63]	ResNet-50 ×4	-	-	**91.2**	-	95.0
Methods using representation learning only:						
Instance Discrimination [60]	ResNet-50	39.2	-	77.4	-	-
Rotation [63]	ResNet-152 ×2	57.5	-	86.4	-	-
ResNet on BigBiGAN (fixed)	RevNet-50 ×4	55.2	73.7	78.8	85.5	87.0
ResNet on AMDIM (fixed)	Custom-103	67.4	81.8	85.8	91.0	92.2
ResNet on CPC v2 (fixed)	ResNet-161	77.1	87.5	90.5	95.0	96.2
ResNet on CPC v2 (fine-tuned)	ResNet-161	**77.9***	**88.6**	**91.2**	**95.6**$^{\#}$	**96.5**

Momentum Contrast (MoCo)

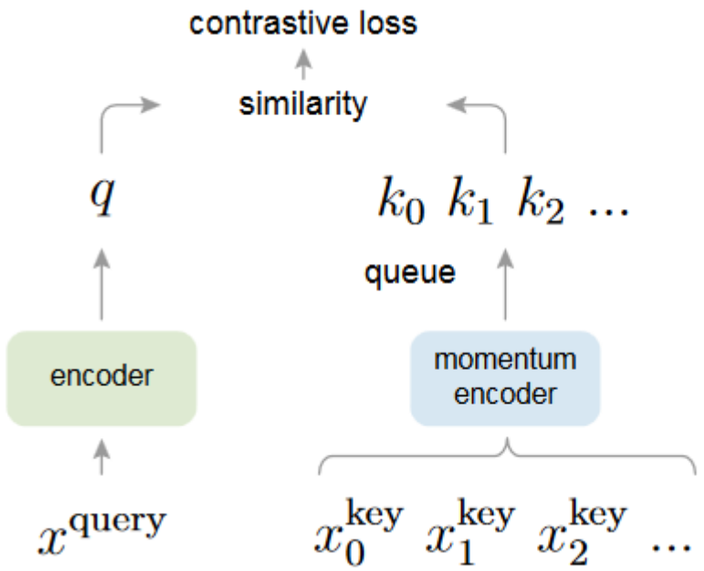


Figure 1. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

если есть запрос *q* и ключи {*k*$_i$} среди которых максимально похожий *k*$_+$, то аналог InfoNCE

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, Ross Girshick «Momentum Contrast for Unsupervised Visual Representation Learning» https://arxiv.org/abs/1911.05722

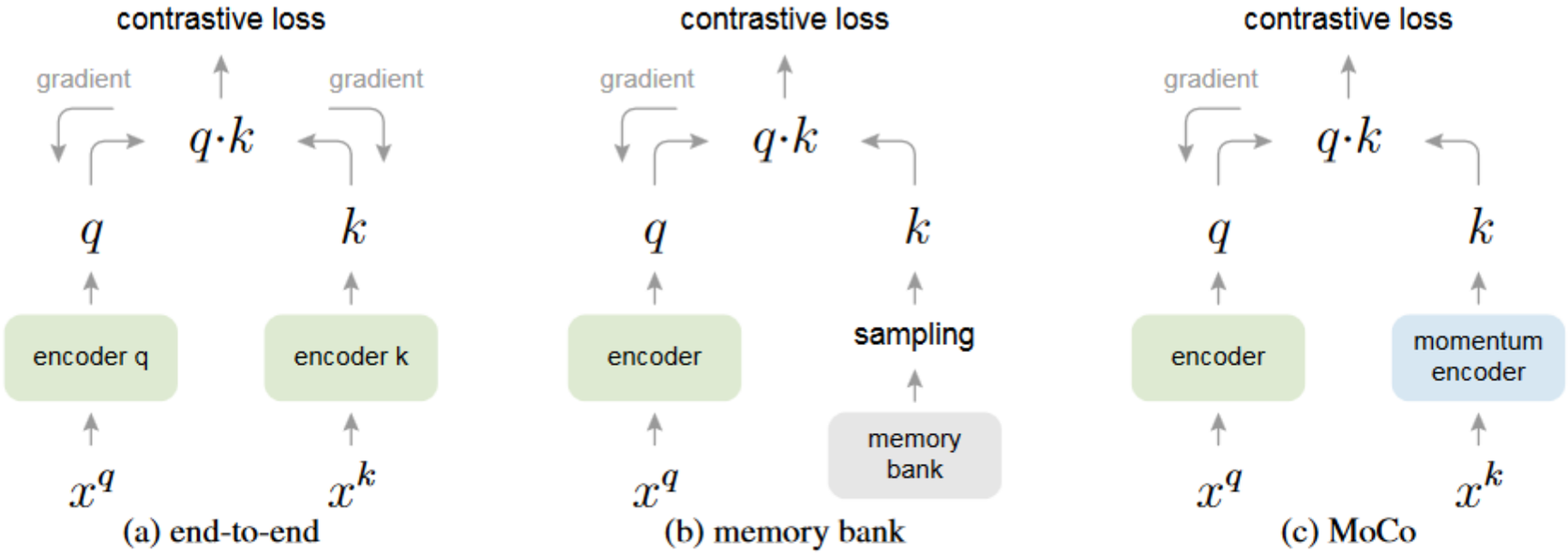


Figure 2. **Conceptual comparison of three contrastive loss mechanisms** (empirical comparisons are in Figure 3 and Table 3). Here we illustrate one pair of query and key. The three mechanisms differ in how the keys are maintained and how the key encoder is updated. **(a)**: The encoders for computing the query and key representations are updated ***end-to-end*** by back-propagation (the two encoders can be different). **(b)**: The key representations are sampled from a ***memory bank*** [61]. **(c)**: ***MoCo*** encodes the new keys on-the-fly by a momentum-updated encoder, and maintains a queue (not illustrated in this figure) of keys.

contrastive learning – способ построения дискретного словаря высокоразмерного непрерывного входа (ex: изображений)

словарь динамичный – ключи сэмплируются, кодировщик ключей эволюционирует

кодировщик – HC ResNet

Pretext Task здесь простая – одно и тоже ли это изображение с точностью до аугментации

Momentum Contrast (MoCo)

одна из главных идей:
чем больше негативных примеров, тем лучше
обычно их число ограничивается батчем

будем поддерживать очередь негативных примеров
(заносим туда очередной батч, а очень старый выносим)
но градиент через них не будем пропускать,
вместо этого momentum update:

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}$$

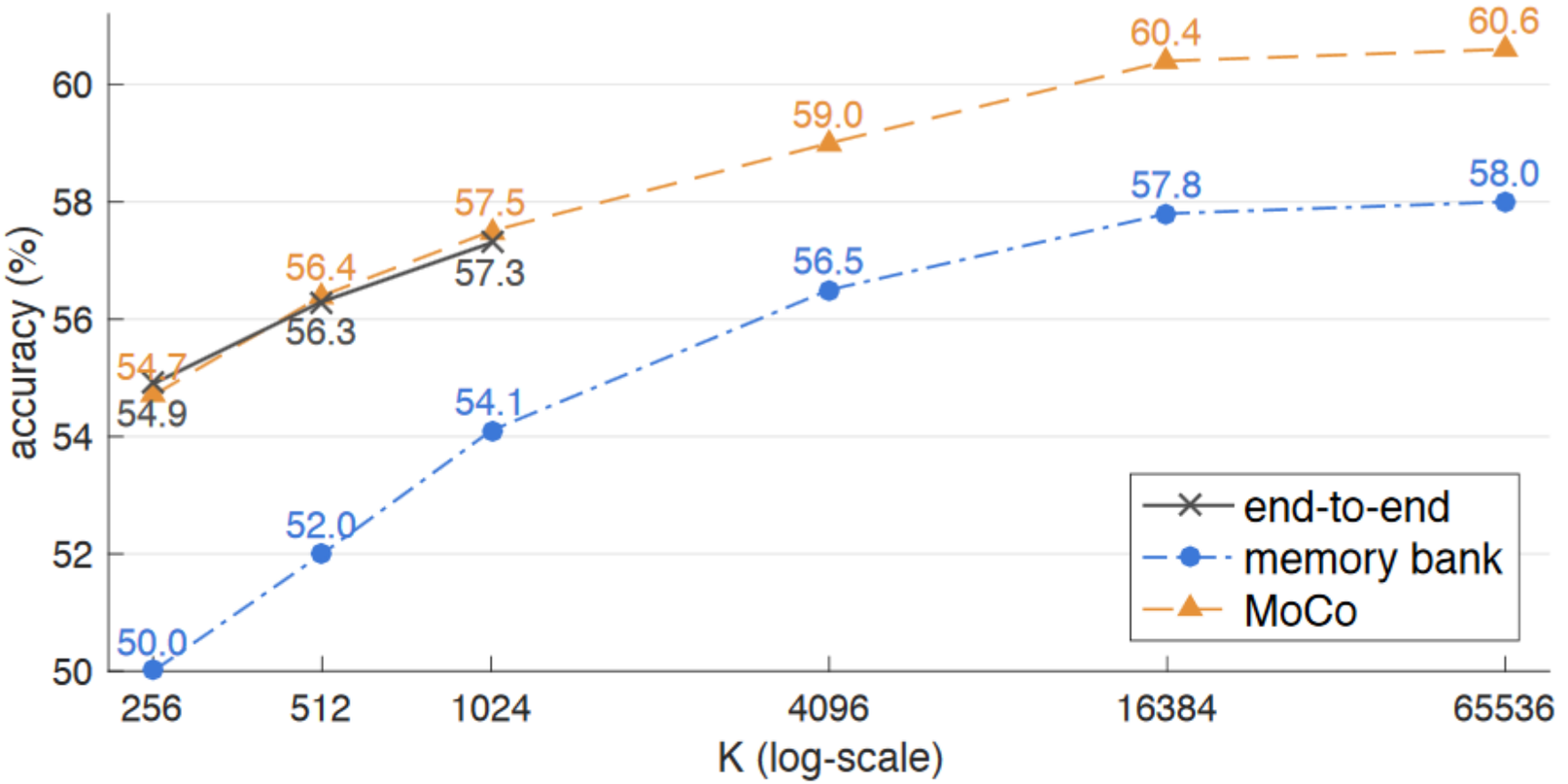


Figure 3. **Comparison of three contrastive loss mechanisms** under the ImageNet linear classification protocol. We adopt the same pretext task (Sec. 3.3) and only vary the contrastive loss mechanism (Figure 2). The number of negatives is K in memory bank and MoCo, and is $K-1$ in end-to-end (offset by one because the positive key is in the same mini-batch). The network is ResNet-50.

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations

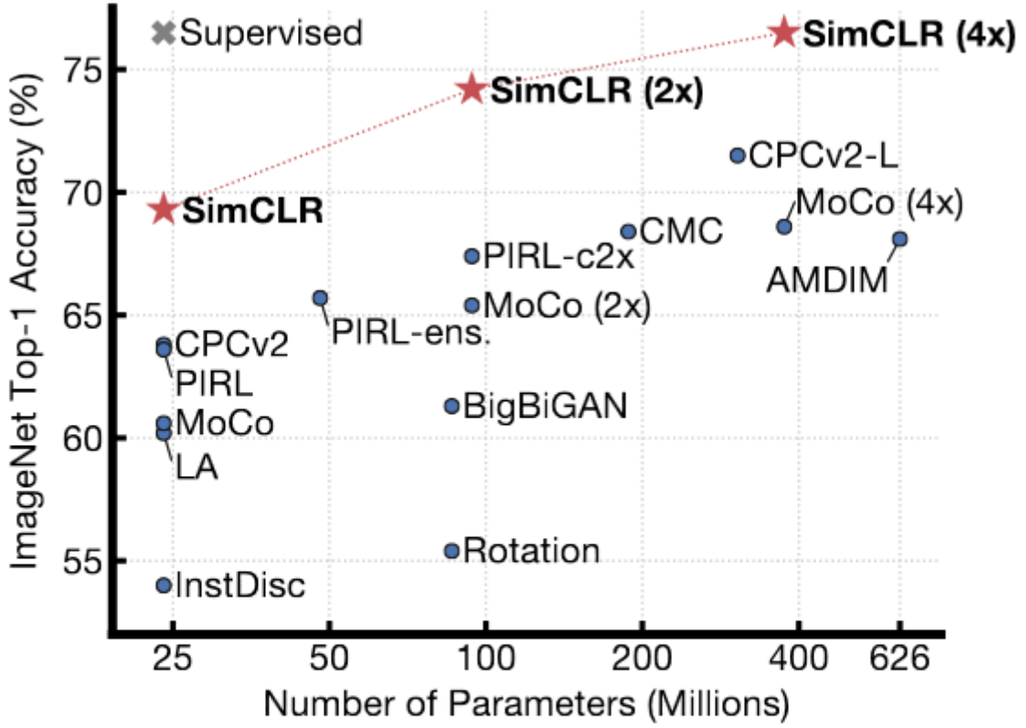


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton «A Simple Framework for Contrastive Learning of Visual Representations» https://arxiv.org/abs/2002.05709

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations

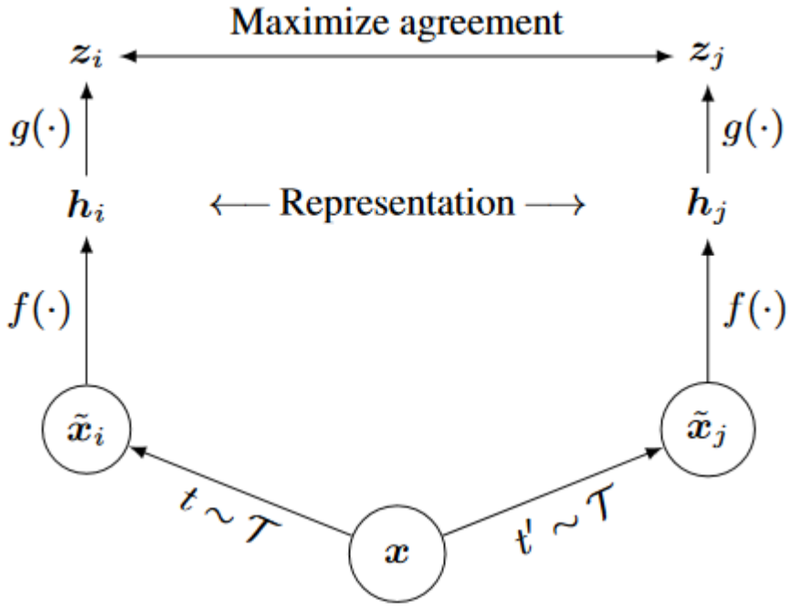


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder 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.

Augmentations

- random cropping + resize (to the original size)
- random color distortions
- random Gaussian blur

base encoder

- ResNet

projection head

$$\boldsymbol{z}_i = g(\boldsymbol{h}_i) = W^{(2)}\sigma(W^{(1)}\boldsymbol{h}_i)$$

Contrastive loss

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations

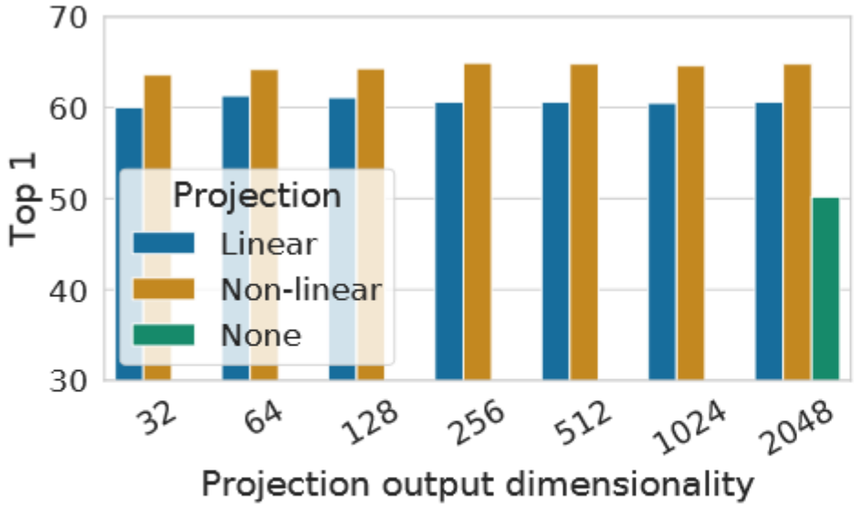


Figure 8. Linear evaluation of representations with different projection heads $g(\cdot)$ and various dimensions of $\boldsymbol{z} = g(\boldsymbol{h})$. The representation $\boldsymbol{h}$ (before projection) is 2048-dimensional here.

What to predict?	Random guess	Representation $\boldsymbol{h}$	Representation $g(\boldsymbol{h})$
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

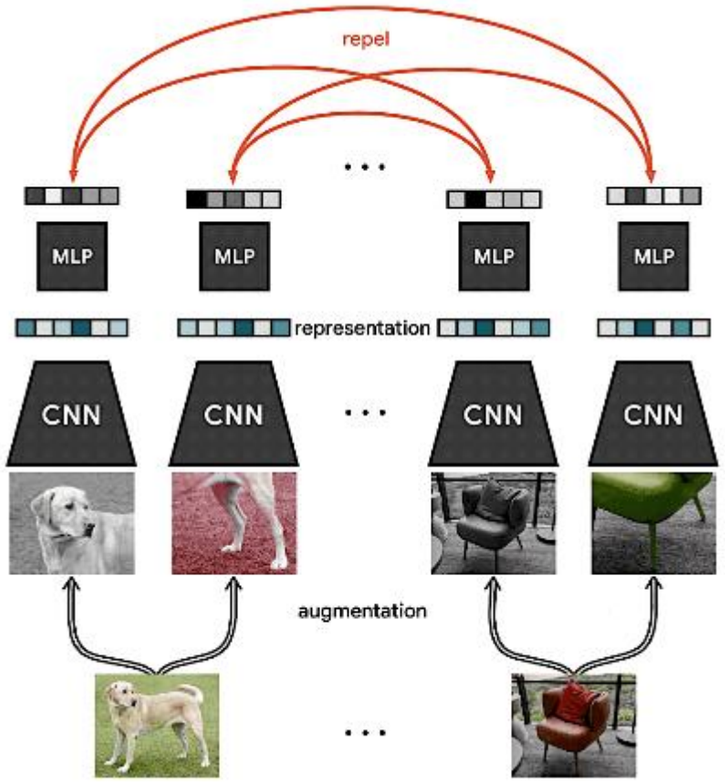
Table 3. Accuracy of training additional MLPs on different representations to predict the transformation applied. Other than crop and color augmentation, we additionally and independently add rotation (one of $\{0°, 90°, 180°, 270°\}$), Gaussian noise, and Sobel filtering transformation during the pretraining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

эксперименты с выбором «projection head»

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of f, g, $\mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \dots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \dots, 2N\}$ and $j \in \{1, \dots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i,j)$ **as** $\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations



берём N изображений, из каждого выделяем 2 аугментированных патча, каждый может выступить как якорное, у него будет 1 позитивный и 2N-2 негативных

SimCLR: A Simple Framework for Contrastive Learning of Visual Representations

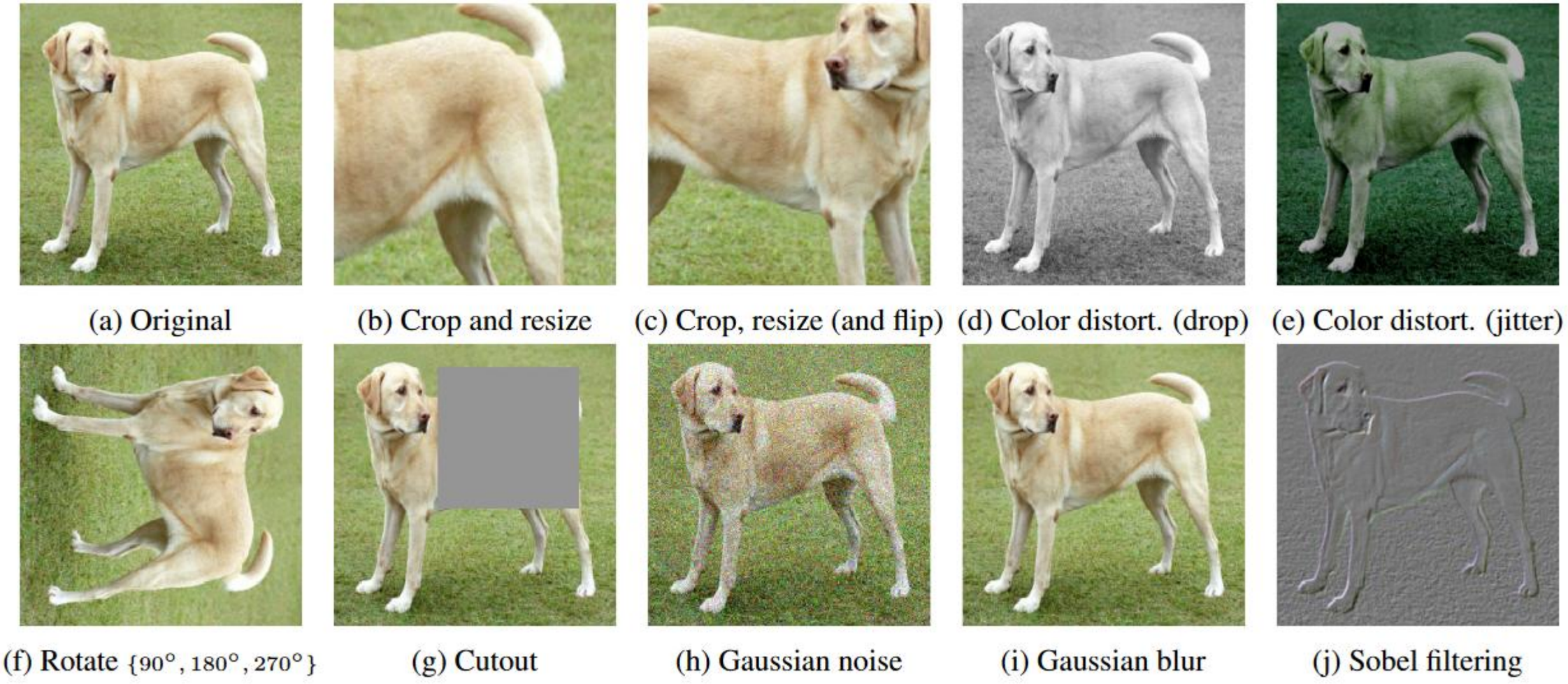


Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we *only* test these operators in ablation, the *augmentation policy used to train our models* only includes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

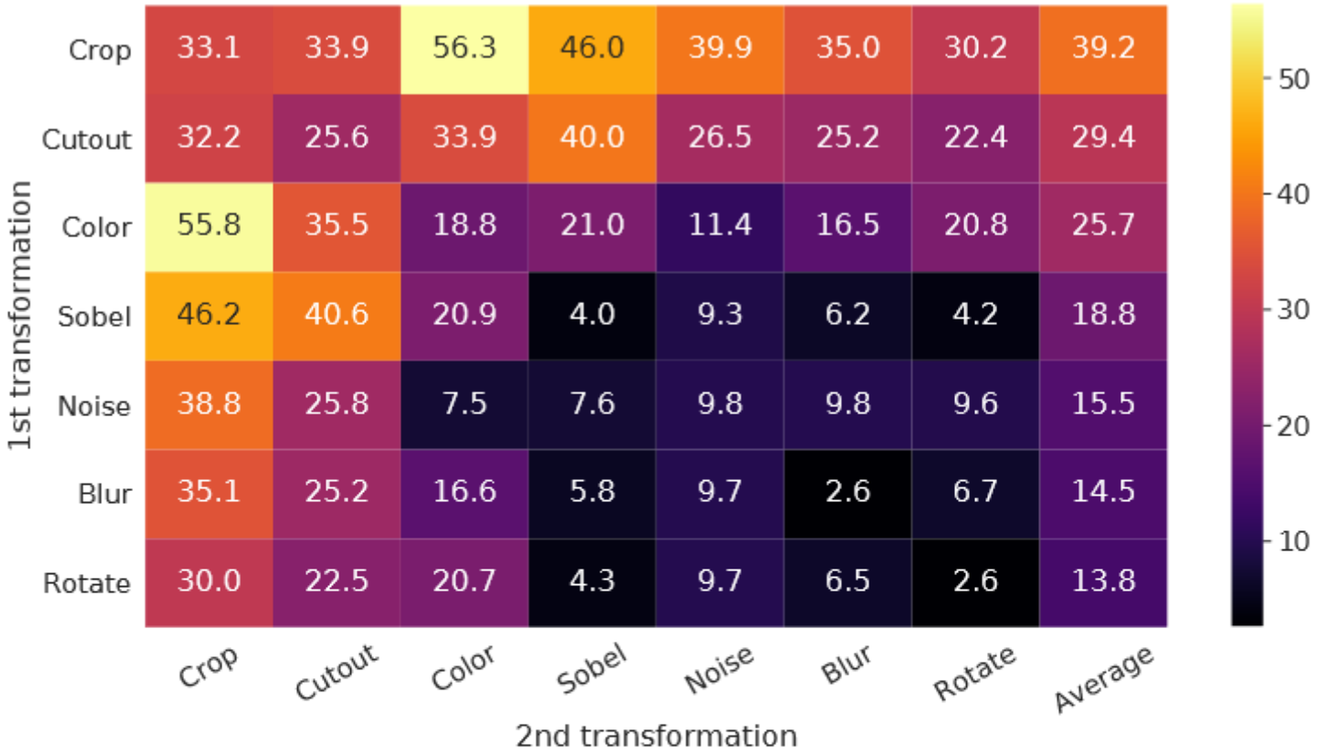


Figure 5. Linear evaluation (ImageNet top-1 accuracy) under individual or composition of data augmentations, applied only to one branch. For all columns but the last, diagonal entries correspond to single transformation, and off-diagonals correspond to composition of two transformations (applied sequentially).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

Method	Architecture	Param.	Top 1	Top 5
Methods using ResNet-50:				
Local Agg.	ResNet-50	24	60.2	-
MoCo	ResNet-50	24	60.6	-
PIRL	ResNet-50	24	63.6	-
CPC v2	ResNet-50	24	63.8	85.3
SimCLR (ours)	ResNet-50	24	**69.3**	**89.0**
Methods using other architectures:				
Rotation	RevNet-50 (4×)	86	55.4	-
BigBiGAN	RevNet-50 (4×)	86	61.3	81.9
AMDIM	Custom-ResNet	626	68.1	-
CMC	ResNet-50 (2×)	188	68.4	88.2
MoCo	ResNet-50 (4×)	375	68.6	-
CPC v2	ResNet-161 (∗)	305	71.5	90.1
SimCLR (ours)	ResNet-50 (2×)	94	74.2	92.0
SimCLR (ours)	ResNet-50 (4×)	375	**76.5**	**93.2**

Table 6. ImageNet accuracies of linear classifiers trained on representations learned with different self-supervised methods.

Method	Architecture	Label fraction (Top 5) 1%	10%
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet-50	39.2	77.4
BigBiGAN	RevNet-50 (4×)	55.2	78.8
PIRL	ResNet-50	57.2	83.8
CPC v2	ResNet-161(∗)	77.9	91.2
SimCLR (ours)	ResNet-50	75.5	87.8
SimCLR (ours)	ResNet-50 (2×)	83.0	91.2
SimCLR (ours)	ResNet-50 (4×)	**85.8**	**92.6**

Table 7. ImageNet accuracy of models trained with few labels.

MoCo v2

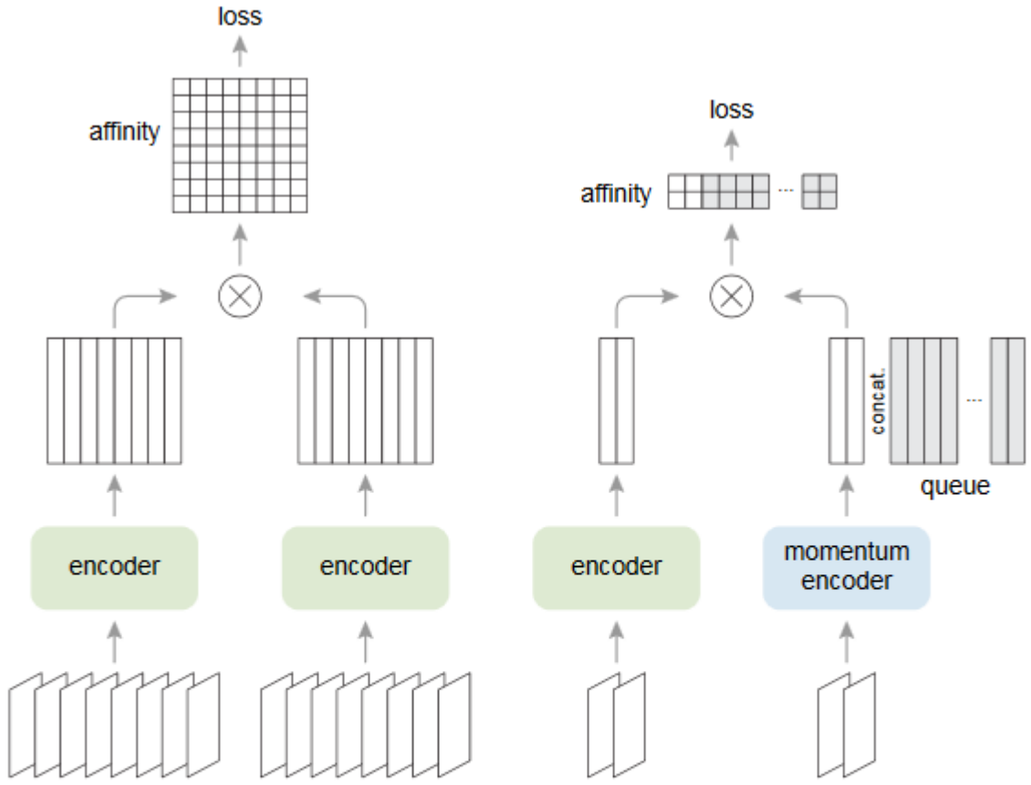


Figure 1. A **batching** perspective of two optimization mechanisms for contrastive learning. Images are encoded into a representation space, in which pairwise affinities are computed.

Xinlei Chen, Haoqi Fan, Ross Girshick, Kaiming He «Improved Baselines with Momentum Contrastive Learning» https://arxiv.org/abs/2003.04297

MoCo v2

case	unsup. pre-train MLP	aug+	cos	epochs	batch	ImageNet acc.
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	**67.5**
*results of **longer** unsupervised training follow:*						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	**71.1**

Table 2. **MoCo *vs.* SimCLR**: ImageNet linear classifier accuracy (**ResNet-50, 1-crop 224×224**), trained on features from unsupervised pre-training. "aug+" in SimCLR includes blur and stronger color distortion. SimCLR ablations are from Fig. 9 in [2] (we thank the authors for providing the numerical results).

fc head in MoCo → a 2-layer MLP head
+ blur augmentation

Video Noise Contrastive Estimation

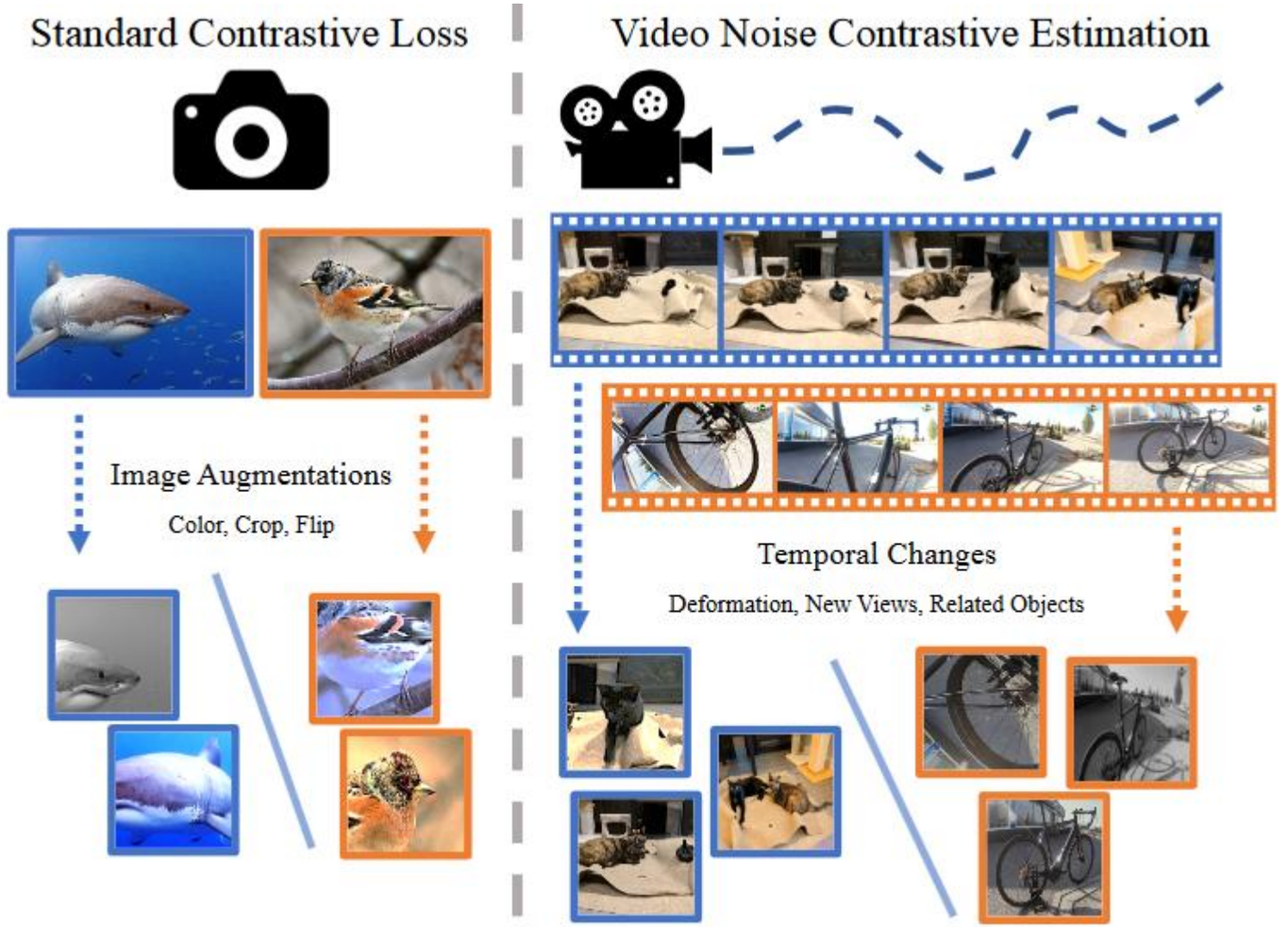
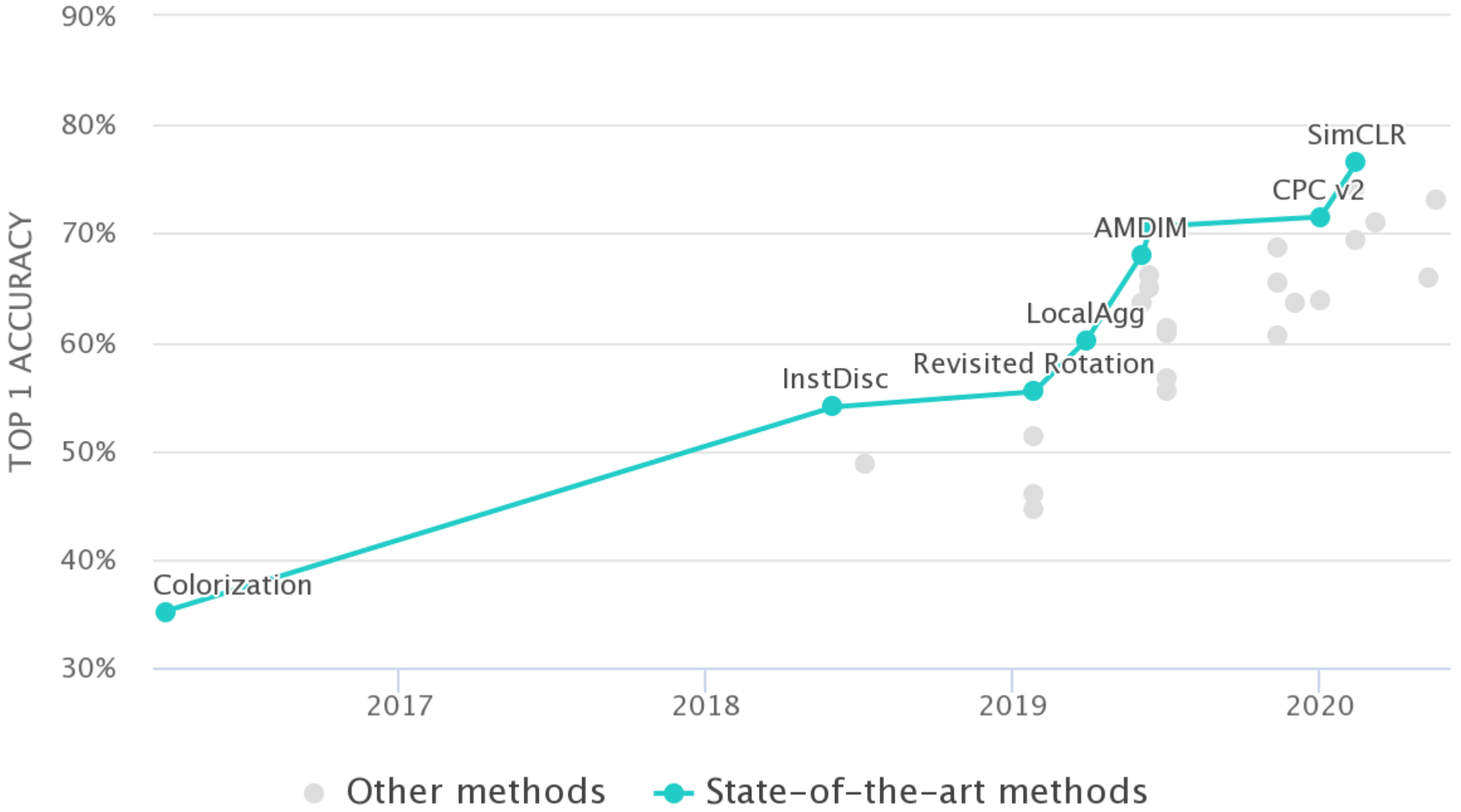


Fig. 1: The standard unsupervised learning setup learns to separate multiple augmentations of the same image. Our method uses truly novel views and temporal consistency which single images cannot provide.

Daniel Gordon «Watching the World Go By:Representation Learning from Unlabeled Videos» // https://arxiv.org/pdf/2003.07990.pdf



https://paperswithcode.com/sota/self-supervised-image-classification-on

Исследование архитектур для самообучения

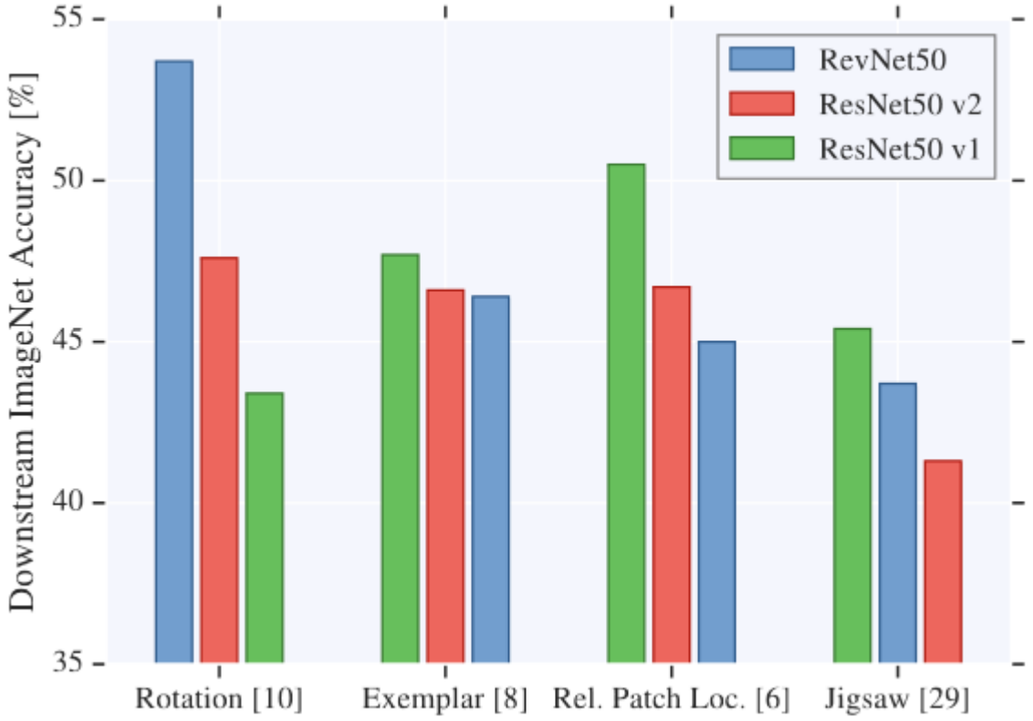


Figure 1. Quality of visual representations learned by various self-supervised learning techniques significantly depends on the convolutional neural network architecture that was used for solving the self-supervised learning task. In our paper we provide a large scale in-depth study in support of this observation and discuss its implications for evaluation of self-supervised models.

Kolesnikov A., Zhai X., Beyer L. Revisiting self-supervised visual representation learning //Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. – 2019. – С. 1920-1929. https://arxiv.org/pdf/1901.09005.pdf

Исследование архитектур для самообучения

Table 1. Evaluation of representations from self-supervised techniques based on various CNN architectures. The scores are accuracies (in %) of a linear logistic regression model trained on top of these representations using *ImageNet* training split. Our validation split is used for computing accuracies. The architectures marked by a "(-)" are slight variations described in Section 3.1. Sub-columns such as 4× correspond to widening factors. Top-performing architectures in a column are bold; the best pretext task for each model is underlined.

Model	Rotation				Exemplar			RelPatchLoc		Jigsaw	
	4×	8×	12×	16×	4×	8×	12×	4×	8×	4×	8×
RevNet50	**47.3**	**50.4**	**53.1**	**53.7**	**42.4**	45.6	46.4	40.6	45.0	40.1	43.7
ResNet50 v2	43.8	47.5	47.2	47.6	**43.0**	45.7	46.6	42.2	46.7	38.4	41.3
ResNet50 v1	41.7	43.4	43.3	43.2	**42.8**	**46.9**	**47.7**	**46.8**	**50.5**	**42.2**	**45.4**
RevNet50 (-)	45.2	**51.0**	**52.8**	**53.7**	38.0	42.6	44.3	33.8	43.5	36.1	41.5
ResNet50 v2 (-)	38.6	44.5	47.3	48.2	33.7	36.7	38.2	38.6	43.4	32.5	34.4
VGG19-BN	16.8	14.6	16.6	22.7	26.4	28.3	29.0	28.5	29.4	19.8	21.1

Исследование архитектур для самообучения

Family		ImageNet		Places205	
		Prev.	Ours	Prev.	Ours
A	Rotation[11]	38.7	**55.4**	35.1	**48.0**
R	Exemplar[8]	31.5	46.0	-	42.7
R	Rel. Patch Loc.[8]	36.2	51.4	-	45.3
A	Jigsaw[34, 51]	34.7	44.6	35.5	42.2
V	CC+vgg-Jigsaw++[36]	37.3	-	37.5	-
A	Counting[35]	34.3	-	36.3	-
A	Split-Brain[51]	35.4	-	34.1	-
V	DeepClustering[3]	**41.0**	-	**39.8**	-
R	CPC[37]	48.7†	-	-	-
R	Supervised RevNet50	74.8	74.4	-	58.9
R	Supervised ResNet50 v2	76.0	75.8	-	61.6
V	Supervised VGG19	72.7	75.0	58.9	61.5

† *marks results reported in unpublished manuscripts.*

Table 2. Comparison of the published self-supervised models to our best models. The scores correspond to accuracy of linear logistic regression that is trained on top of representations provided by self-supervised models. Official validation splits of *ImageNet* and *Places205* are used for computing accuracies. The "Family" column shows which basic model architecture was used in the referenced literature: **A**lexNet, **V**GG-style, or **R**esidual.

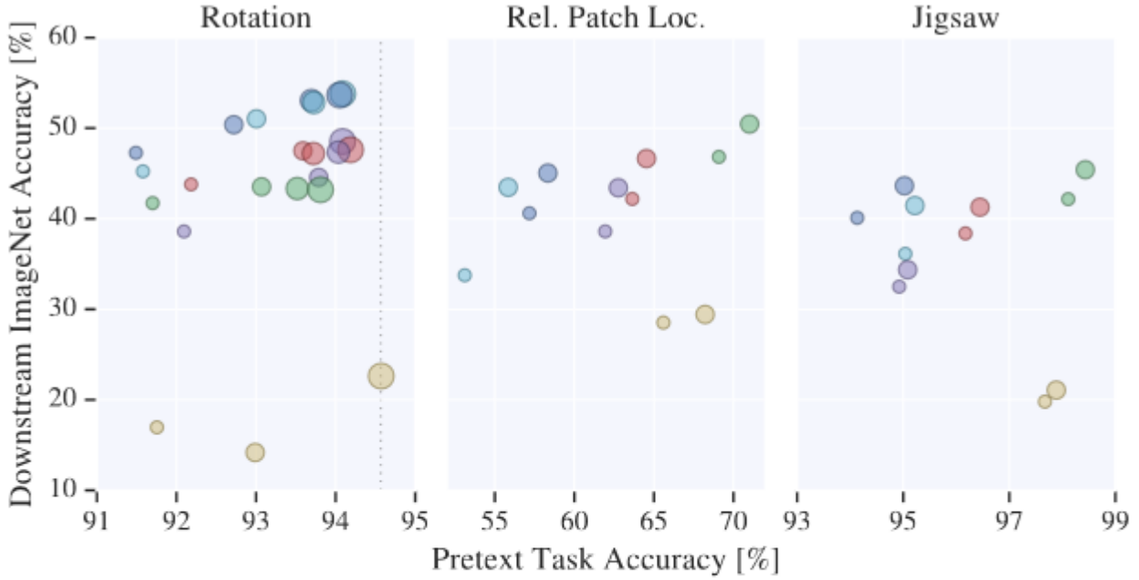


Figure 4. A look at how predictive pretext performance is to eventual downstream performance. Colors correspond to the architectures in Figure 3 and circle size to the widening factor k. Within an architecture, pretext performance is somewhat predictive, but it is not so across architectures. For instance, according to pretext accuracy, the widest *VGG* model is the best one for *Rotation*, but it performs poorly on the downstream task.

не всегда есть связь качества решения на предварительной и окончательной задачах лишь в рамках одной архитектуры

Исследование архитектур для самообучения

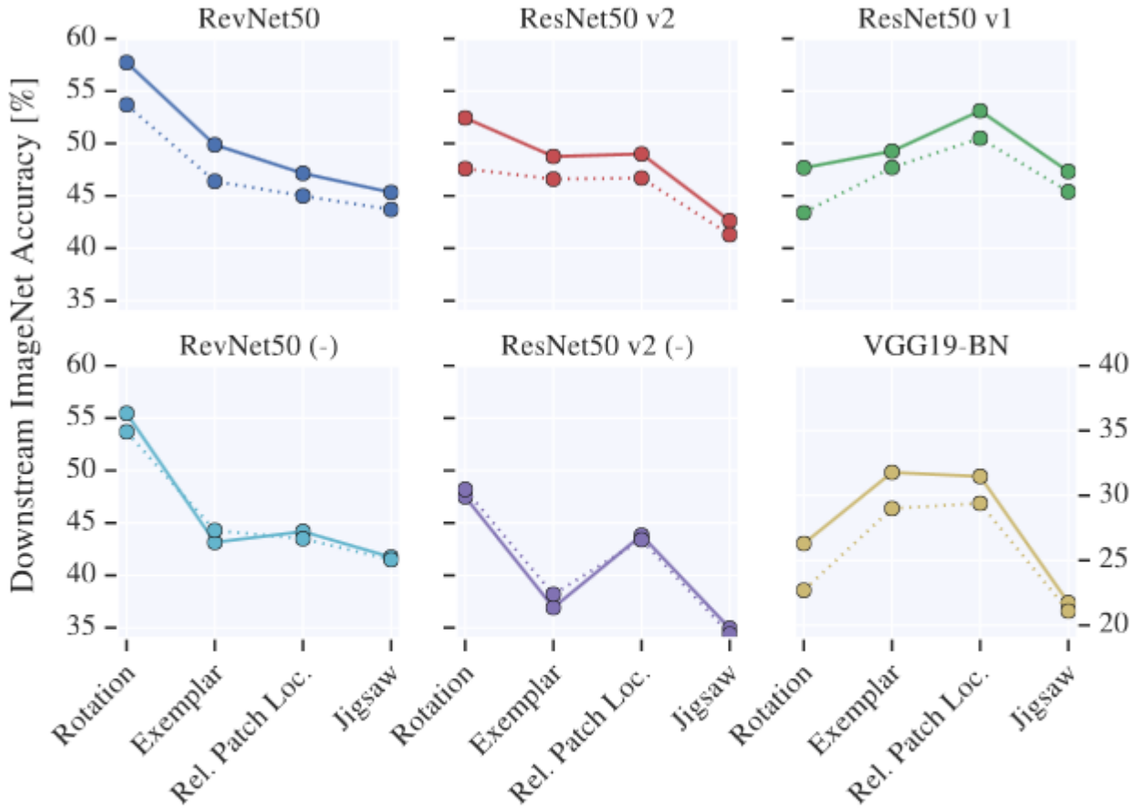


Figure 3. Comparing linear evaluation (·····) of the representations to non-linear (——) evaluation, *i.e.* training a multi-layer perceptron instead of a linear model. Linear evaluation is not limiting: conclusions drawn from it carry over to the non-linear evaluation.

если использовать более сложную модель на полученных представлениях

Исследование архитектур для самообучения

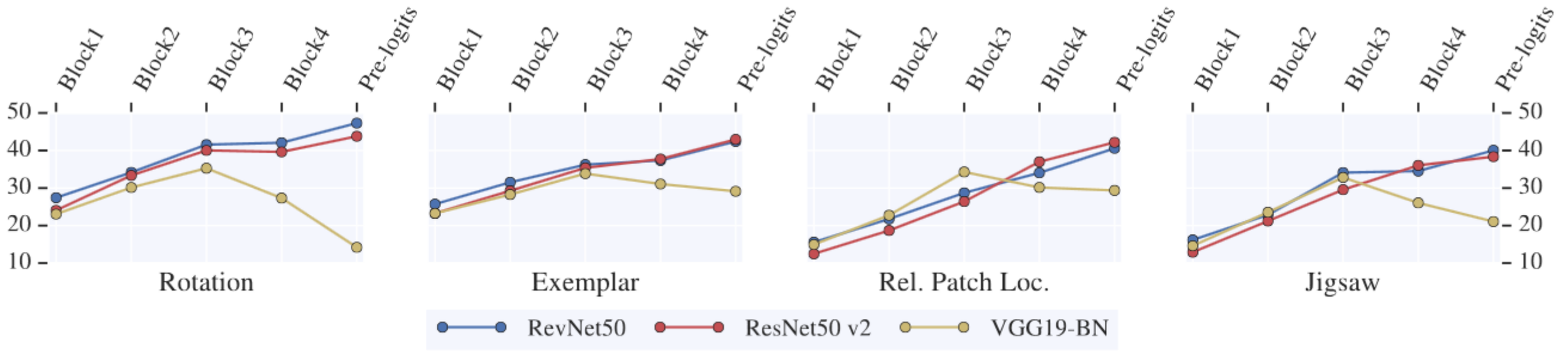


Figure 5. Evaluating the representation from various depths within the network. The vertical axis corresponds to downstream ImageNet performance in percent. For residual architectures, the *pre-logits* are always best.

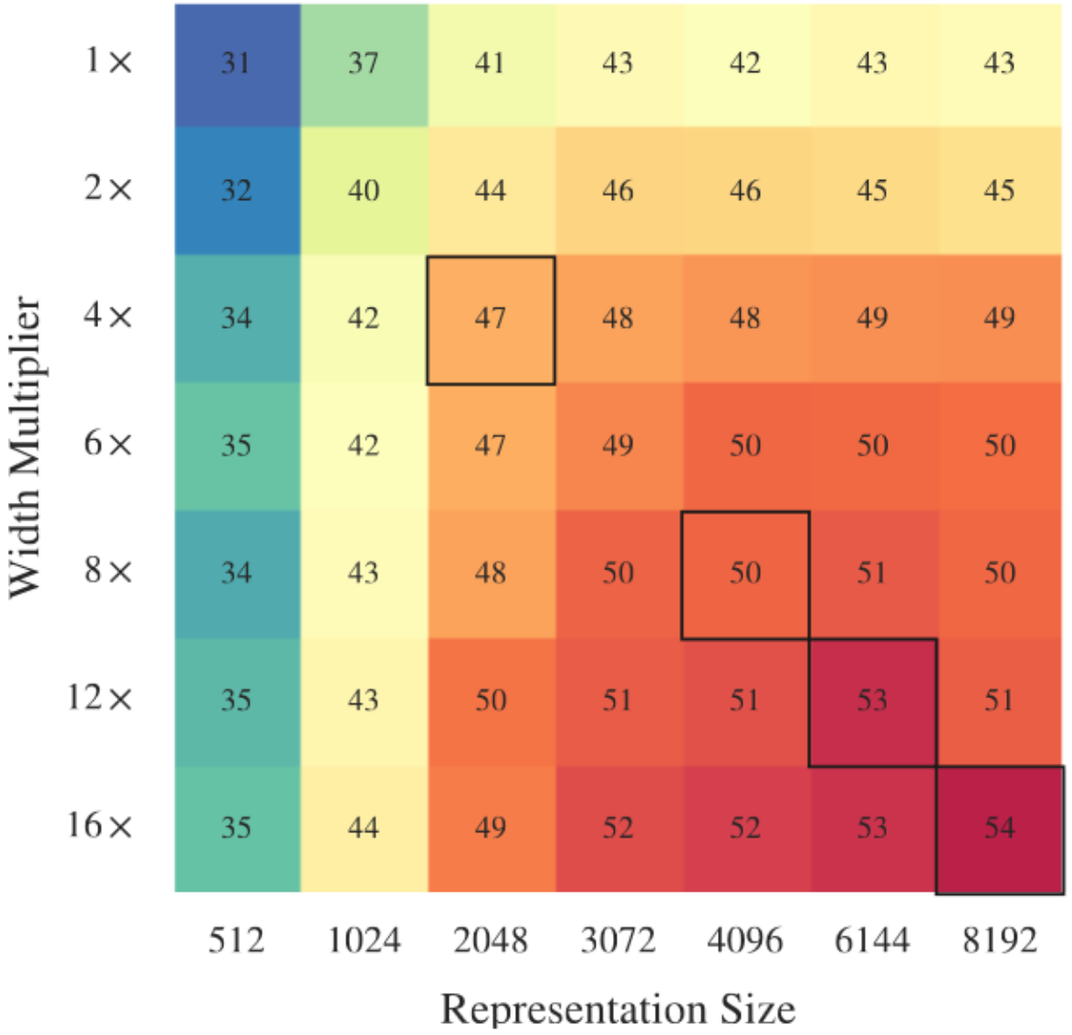


Figure 6. Disentangling the performance contribution of network widening factor versus representation size. Both matter independently, and larger is always better. Scores are accuracies of logistic regression on *ImageNet*. Black squares mark models which are also present in Table 1.

Исследование архитектур для самообучения

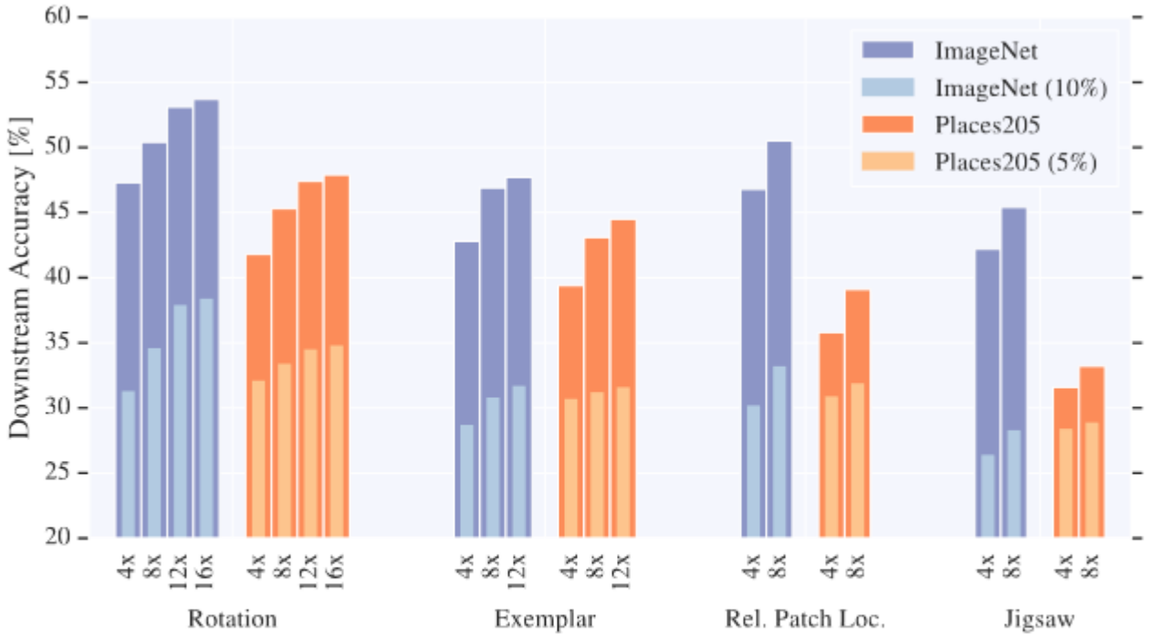


Figure 7. Performance of the best models evaluated using all data as well as a subset of the data. The trend is clear: increased widening factor increases performance across the board.

Некоторые кейсы

Human Activity Recognition

Проблема – дорогие и неанонимные размеченные данные

learn a multi-stream temporal convolutional network to recognize transformations applied on the input signals –
«Transformation Prediction Network (TPN)»

Noised
Scaled
Rotated
Negated
Flipped
Permuted
Time-Warped
Channel-Shuffled

Aaqib Saeed, Tanir Ozcelebi, Johan Lukkien «Multi-modal Self-Supervised Learning for Human Activity Recognition» // https://drive.google.com/file/d/0B4M2lUVyJzS4WHVLWjdZeGVZLWVDb1puX3N2b19lc0xRQzMw/view

Human Activity Recognition

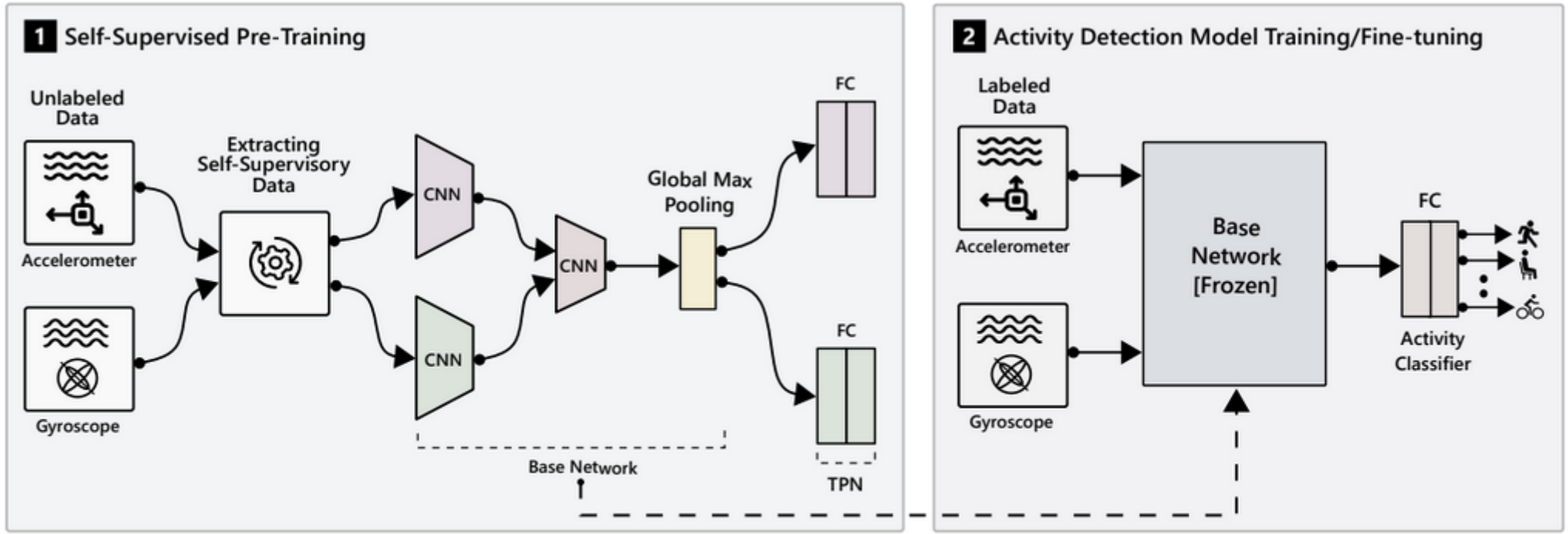


Figure 1. **Illustration of the proposed multi-modal self-supervised approach for sensory feature learning.** We train a temporal convolutional network for transformation recognition as a pretext task as shown in Step 1. The learned features are utilized by the activity recognition model (Step 2) to demonstrate an improved detection rate with a labeled dataset.

Human Activity Recognition

Table 1. **Task Generalization: Evaluating self-supervised representations for activity recognition.**

	HHAR		MobiAct		MotionSense		UCI HAR	
	F-score	Kappa	F-score	Kappa	F-score	Kappa	F-score	Kappa
Random Init.	0.12±0.09	0.08±0.09	0.41±0.11	0.33±0.09	0.15±0.08	0.11±0.08	0.33±0.06	0.35±0.07
Supervised	0.80±0.03	0.77±0.04	0.94±0.01	0.92±0.01	0.92±0.02	0.91±0.02	0.94±0.01	0.92±0.01
Autoencoder	0.80±0.02	0.76±0.02	0.86±0.01	0.82±0.01	0.88±0.01	0.85±0.01	0.89±0.02	0.87±0.02
Self-Supervised	0.87±0.01	0.84±0.01	0.93±0.00	0.91±0.01	0.92±0.01	0.89±0.01	0.94±0.00	0.93±0.00
Self-Supervised (FT)	0.86±0.01	0.83±0.01	0.94±0.01	0.92±0.01	0.94±0.01	0.93±0.01	0.96±0.01	0.95±0.01

FT – последний слой сети fine-tuned

Paired Cell Inpainting

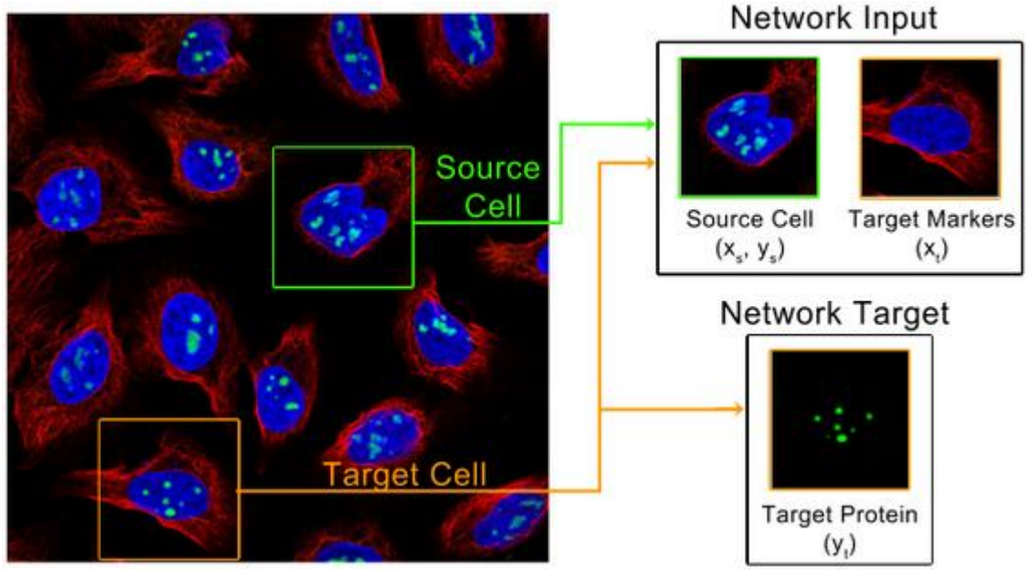


Figure 1. Inputs and targets to the network. We crop a source cell (green border) and a target cell (orange border) from the same image. Then, given all channels for the source cell, and the structural markers for the target cell (in this dataset, the nucleus and the microtubule channels), the network is trained to predict the appearance of the protein channel in the target cell. Images shown are of human cells, with the nucleus colored blue, microtubules colored red, and a protein localized to the nucleoli colored green.

цель – обучить представление отдельной клетки
(на изображении их несколько)
здесь таргетное изображение в других каналах (в которых протеины изображены)

Alex X. Lu, Amy X. Lu, Oren Z. Kraus, Sam Cooper, Wiebke Schormann, David W. Andrews, Alan M. Moses «Paired Cell Inpainting: Self-Supervised Multiple-Instance Learning for Bioimage Analysis» // https://drive.google.com/file/d/0B4M2lUVyJzS4d3B2X3AtVmxFTm5iT0dmN2RrVmhCRFY4MGdj/view

Paired Cell Inpainting (PCI)

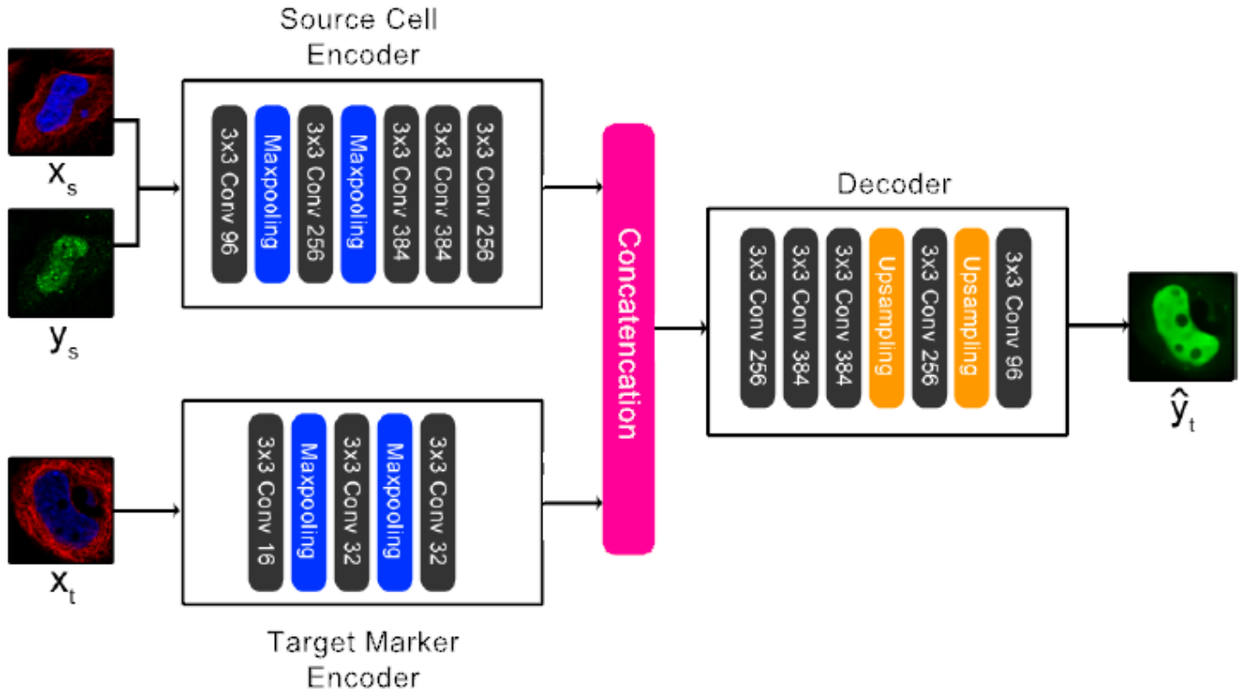


Figure 2. A summary of our architecture. Our architecture consists of a source cell encoder and a target marker encoder. The final layers of both encoders are concatenated, and fed into a decoder that outputs the prediction of the target protein $\hat{y}_t$. We show a real example of a prediction from our trained human cell model given the input image patches in this schematic.

Paired Cell Inpainting (PCI)

Table 1. Summary of datasets used to train models.

Dataset	Channels	Images	Single Cell Crops
Yeast	2	4,069	1,165,713
Human	3	41,285	638,640
Mouse	2	3,600	58,031

Table 3. Classification accuracies for various feature sets.

Features	Unsupervised?	Accuracy
Texture	√	68.15
Engineered	√	62.04
Autoencoder	√	42.50
VGG16	√	69.33
PCI	√	**87.98**
Supervised	×	92.44

Classifying protein localization in yeast single cells

Table 2. Balanced classification accuracies for classifiers built for each layer of the yeast paired cell inpainting model.

Layer	Accuracy
Conv1	43.09
Conv2	70.52
Conv3	84.30
Conv4	**87.98**
Conv5	81.89

Table 4. Distance scores for various feature sets.

Feature Set	Distance Score
Texture	-0.2737
Autoencoder	-0.2813
VGG16	-0.3254
PCI (Conv3)	**-0.5743**
PCI (Conv4)	-0.3488

Distances between similar human proteins

Audio2Vec

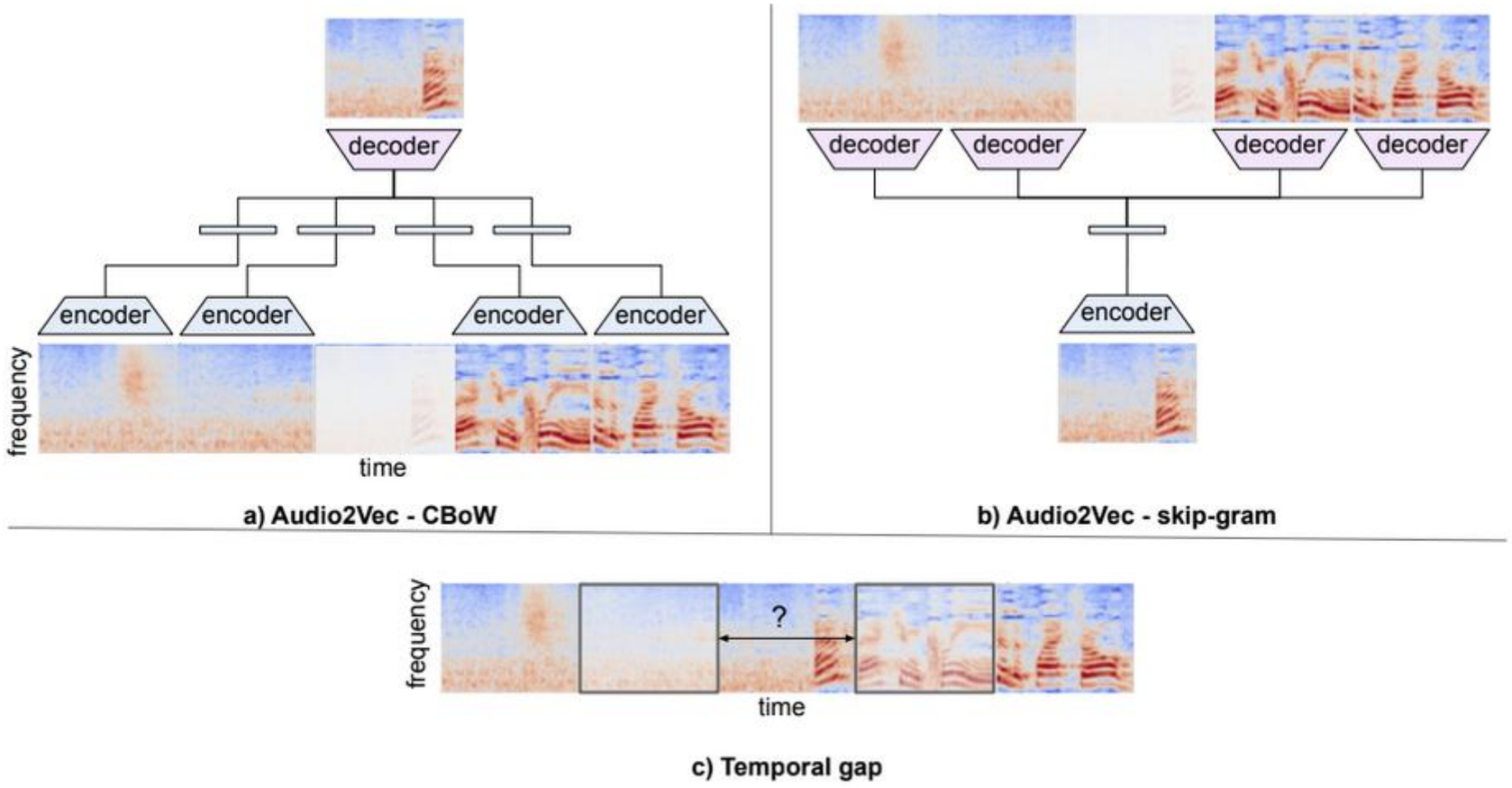


Figure 1. Self-supervised learning tasks: a) Continuous-Bag-of-Words variant of Audio2Vec, b) Skip-gram variant of Audio2Vec, c) Temporal gap task setup.

M. Tagliasacchi, B. Gfeller, D. Roblek «Self-supervised audio representation learning based on temporal context»
https://drive.google.com/file/d/0B4M2IUVyJzS4YlRnaDl6TUhGZG51STFaZno0R2pwd09IUUIn/view

Audio2Vec

model	SC1	SC2	LSP	TUT	MUS	BSD	LID
Spectrogram	0% (0.63)	0% (0.04)	0% (0.00)	0% (0.37)	0% (0.77)	0% (0.67)	0% (0.37)
Untrained	0% (0.63)	6% (0.08)	6% (0.07)	8% (0.41)	54% (0.89)	-19% (0.66)	47% (0.50)
AutoEncoder	25% (0.71)	20% (0.17)	71% (0.71)	26% (0.49)	45% (0.87)	-7% (0.67)	56% (0.53)
TemporalGap	12% (0.67)	19% (0.17)	63% (0.63)	35% (0.53)	86% 0.95)	43% (0.69)	67% (0.56)
Audio2Vec (CBoW)	22% (0.71)	14% (0.13)	79% (0.79)	38% (0.55)	94% (0.97)	56% (0.70)	54% (0.52)
Audio2Vec (skip-gram)	17% (0.69)	13% (0.13)	85% (0.84)	43% (0.57)	95% (0.97)	69% (0.71)	33% (0.46)
MultiHead	93% (0.94)	93% (0.65)	96% (0.95)	90% (0.78)	75% (0.93)	147% (0.75)	78% (0.59)
Supervised	100% (0.97)	100% (0.70)	100% (0.99)	100% (0.83)	100% (0.98)	100% (0.72)	100% (0.66)

Table 1. Accuracy on downstream tasks, both as a fraction of accuracy wrt. baselines, and as absolute accuracy. Downstream tasks: SCx (*Speech Commands*), LSP (*LibriSpeech*), TUT *TUT Urban Acoustic Scenes 2018*, MUS *MUSAN*, BSD *Bird Audio Detection*, LID *Spoken Language Identification*.

везде – классификация, нормировка от 0 до 100%

Audio2Vec –CBoW / skip-gram (our)

Temporal Gap (our)

AutoEncoder (the same encoder and decoder architectures)

Spectrogram (receives directly the flattened spectrogram features as input)

Untrained – computes the embeddings with the very same encoder, but using randomly initialized weights

Supervised – task-specific fully supervised model, the same encoder, but trained end-to-end

MultiHead –single shared encoder is composed with a different fully connected layer for each downstream task (trained end-to-end on the average accuracy of the downstream tasks)

Method	Category	Code	Contribution
GAN [83]	Generation	✓	Forerunner of GAN
DCGAN [120]	Generation	✓	Deep convolutional GAN for image generation
WGAN [121]	Generation	✓	Proposed WGAN which makes the training of GAN more stable
BiGAN [122]	Generation	✓	Bidirectional GAN to project data into latent space
SelfGAN [123]	Multiple	✗	Use rotation recognition and GAN for self-supervised learning
ColorfulColorization [18]	Generation	✓	Posing image colorization as a classification task
Colorization [82]	Generation	✓	Using image colorization as the pretext task
AutoColor [124]	Generation	✓	Training ConvNet to predict per-pixel color histograms
Split-Brain [42]	Generation	✓	Using split-brain auto-encoder as the pretext task
Context Encoder [19]	Generation	✓	Employing ConvNet to solve image inpainting
CompletNet [125]	Generation	✓	Employing two discriminators to guarantee local and global consistent
SRGAN [15]	Generation	✓	Employing GAN for single image super-resolution
SpotArtifacts [126]	Generation	✓	Learning by recognizing synthetic artifacts in images
ImproveContext [33]	Context	✗	Techniques to improve context based self-supervised learning methods
Context Prediction [41]	Context	✓	Learning by predicting the relative position of two patches from an image
Jigsaw [20]	Context	✓	Image patch Jigsaw puzzle as the pretext task for self-supervised learning
Damaged Jigsaw [89]	Multiple	✗	Learning by solving jigsaw puzzle, inpainting, and colorization together
Arbitrary Jigsaw [88]	Context	✗	Learning with jigsaw puzzles with arbitrary grid size and dimension
DeepPermNet [127]	Context	✓	A new method to solve image patch jigsaw puzzle
RotNet [36]	Context	✓	Learning by recognizing rotations of images
Boosting [34]	Multiple	✗	Using clustering to boost the self-supervised learning methods
JointCluster [128]	Context	✓	Jointly learning of deep representations and image clusters
DeepCluster [44]	Context	✓	Using clustering as the pretext
ClusterEmbegging [129]	Context	✓	Deep embedded clustering for self-supervised learning
GraphConstraint [43]	Context	✓	Learning with image pairs mined with Fisher Vector
Ranking [38]	Context	✓	Learning by ranking video frames with a triplet loss
PredictNoise [46]	Context	✓	Learning by mapping images to a uniform distribution over a manifold
MultiTask [32]	Multiple	✓	Using multiple pretext tasks for self-supervised feature learning
Learning2Count [130]	Context	✓	Learning by counting visual primitive
Watching Move [81]	Free Semantic Label	✓	Learning by grouping pixels of moving objects in videos
Edge Detection [81]	Free Semantic Label	✓	Learning by detecting edges
Cross Domain [81]	Free Semantic Label	✓	Utilizing synthetic data and its labels rendered by game engines

https://arxiv.org/pdf/1902.06162.pdf

Почему часто используют AlexNet

чтобы сравниваться с предыдущими статьями
с другими сетями не все приёмы самообучения проходили

Итог

пока не работают с «большими изображениями»

большая ручная разметка данных становится менее нужной

часто используется комбинация техник

можно / нужно использовать в прикладных проектах

есть много тонкостей, «чтобы работало»

Хорошие обзоры

Schmarje L. et al. A survey on Semi-, Self-and Unsupervised Techniques in Image Classification //arXiv preprint arXiv:2002.08721. – 2020. https://arxiv.org/pdf/2002.08721.pdf

Longlong Jing and Yingli Tian «Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey» https://arxiv.org/pdf/1902.06162.pdf

Guo-Jun Qi, Jiebo Luo «Small Data Challenges in Big Data Era: A Survey of Recent Progress on Unsupervised and Semi-Supervised Methods» https://arxiv.org/abs/1903.11260

https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

Коллекции статей

https://github.com/jason718/awesome-self-supervised-learning

https://github.com/Sungman-Cho/Awesome-Self-Supervised-Papers