Towards Bayesian Generative Architectures for Out-of-Distribution Tasks

Alan Yuille

Bloomberg Distinguished Professor

Depts. Computer Science and Cognitive Science

Johns Hopkins University

Background.

- There are a range of neural architectures. AlexNet, VGG, ResNet, Transformers, and so on.
- *But I argue that they have big limitations.*
- As a Cognitive Scientist (and as a human with a visual system), I know that the human visual system is much superior to current AI systems (except for a few highly constrained tasks in highly restricted settings).
- When humans see objects, they can recognize it, estimate its 3D orientation, estimate its boundaries, its parts, its attributes, its material properties and the lighting and weather conditions.
- Humans can do this even if the object is heavily occluded, seen out of context and in situations we've never seen it in before.

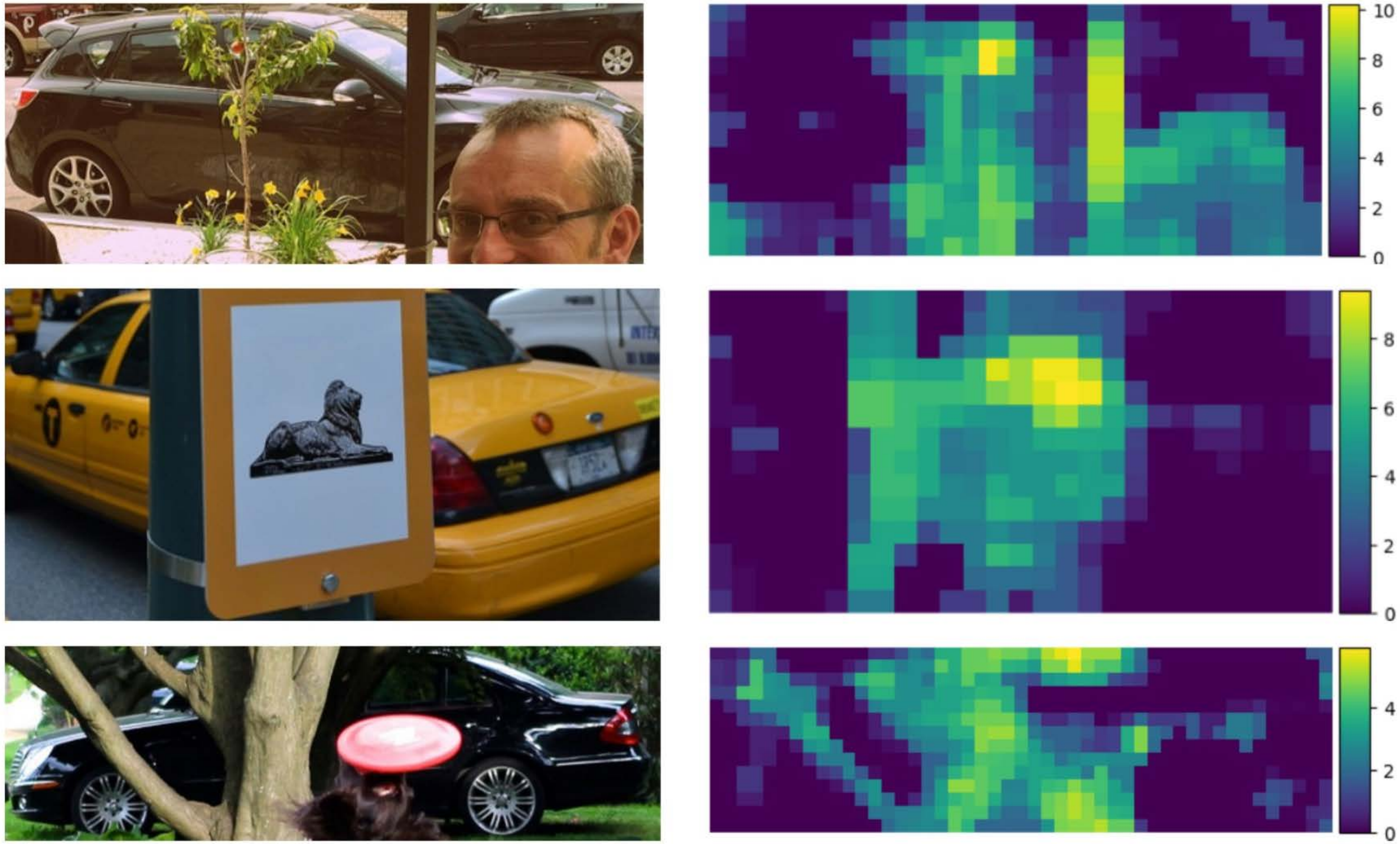
We need tougher ways to test the performance of architectures.

- Standard Performance Measures (SPMs) test algorithms on finite-sized balanced annotated datasets (BAD). Half of the data is used for training and half for testing.
- ***But SPMs are problematic.*** Datasets are biased. They under-represent rare, but important, events. More generally, finite-sized datasets are unable to capture the combinatorial complexity of the real world.
- ***We need tougher tests that can quantify AI vision architectures more realistically.*** Tougher tests include: (I) ***Out-of-DistributionTesting***, where the algorithms are tested from data that has different statistical properties than the training data. (II) ***Adversarial Examiners***, where the examiner selects images adaptively to search for the weaknesses of the algorithms.
- ***We need architectures that can pass these tougher tests.***
- *Alan Yuille & Chenxi Liu: Deep Nets: What have they ever done for Vision? IJCV. 2021.*

Architectures should be Generative and Bayesian

- Current architectures, e.g., CNNs and Transformers. are discriminative.
- *Generative/Bayesian* architectures are more challenging, but offer big advantages particularly for *out-of-distribution* tasks (and are competitive on SPMs).
- This talk starts with a few examples to illustrate this point. The rest of the presentation will present the underlying mathematics of Generative/Bayesian architectures.
- Why Generative? They have knowledge of the generation process: (I) Objects are seen from different viewpoints and have different spatial patterns for each viewpoint. (II) They know that parts of the object are invisible because they are occluded, (III) They can factorize tasks, e.g., enabling domain adaptation.

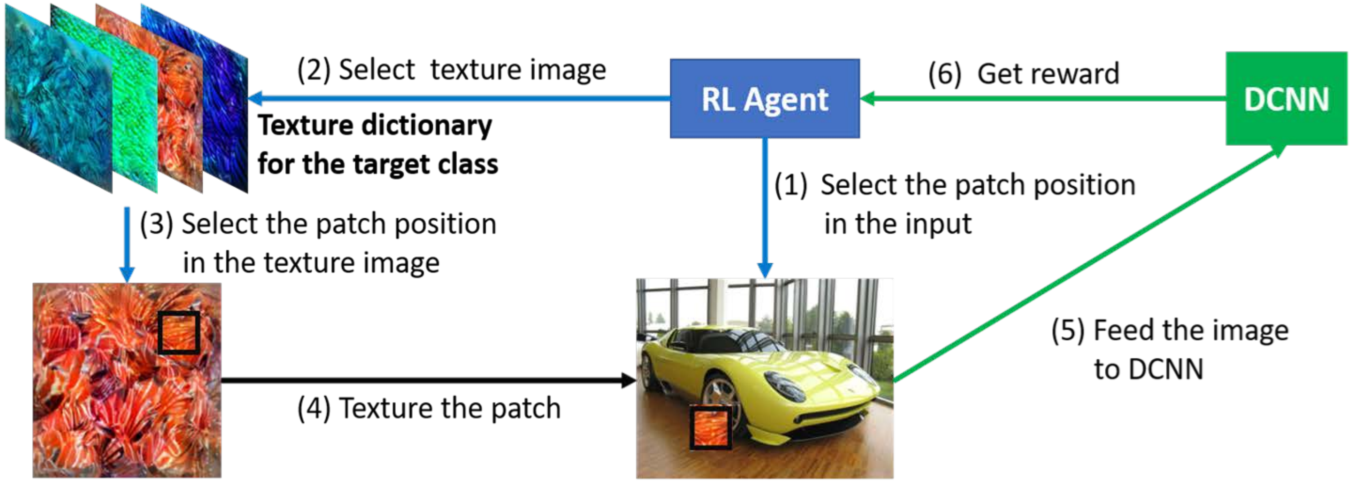
Object Classification and Occlusion Detection



Compositional Generative Nets outperform CNNs when the occlusion levels are high.
A. Kortylewski et al. CVPR 2020, A. Wang et al. CVPR 2020, A. Kortylewski et al. IJCV 2020.

Adversarial Examiners: Patch Attacks

- Learn an Attack Policy by reinforcement learning. (C. Yang et al. ECCV 2020)



(2) Select texture image

RL Agent

(6) Get reward

DCNN

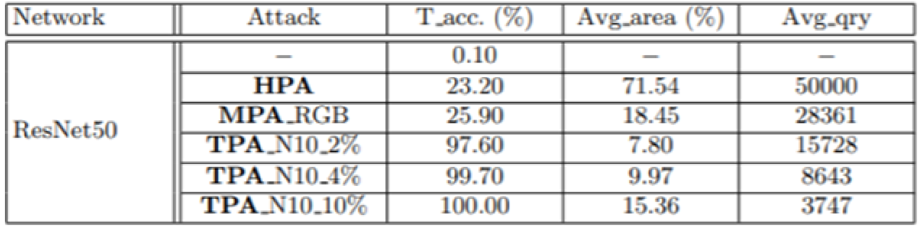
Texture dictionary for the target class

(1) Select the patch position in the input

(3) Select the patch position in the texture image

(5) Feed the image to DCNN

(4) Texture the patch



Network	Attack	T_acc. (%)	Avg_area (%)	Avg_qry
ResNet50	–	0.10	–	–
	HPA	23.20	71.54	50000
	MPA_RGB	25.90	18.45	28361
	TPA_N10_2%	97.60	7.80	15728
	TPA_N10_4%	99.70	9.97	8643
	TPA_N10_10%	100.00	15.36	3747

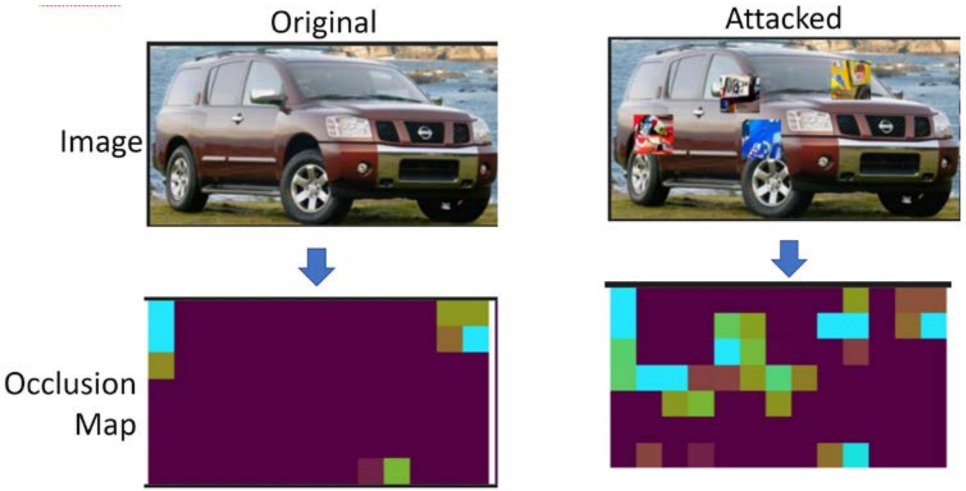
- A black box targeted attack which is almost 100% effective.
- Suggests Deep Nets have little knowledge of the global structure of objects. They are just “bags of patches”. They only pay “attention” to small regions of the image. They find that synthetic texture patches (obtained by a surrogate deep network) look more like an object than the object itself.

CompNets are robust to Patch Attacks.

- CompNets ***are*** robust against targeted patch attacks (unlike CNNs).

Model	Accuracy (%)	Attack success rate (%)	
		PatchAttack[1] (TPA) 4 patches	Sparse-RS[2] **1 patch**
CompNet (vgg16 backbone)	98.5	12.6	0.9
vgg16	98.6	98.8	92.3

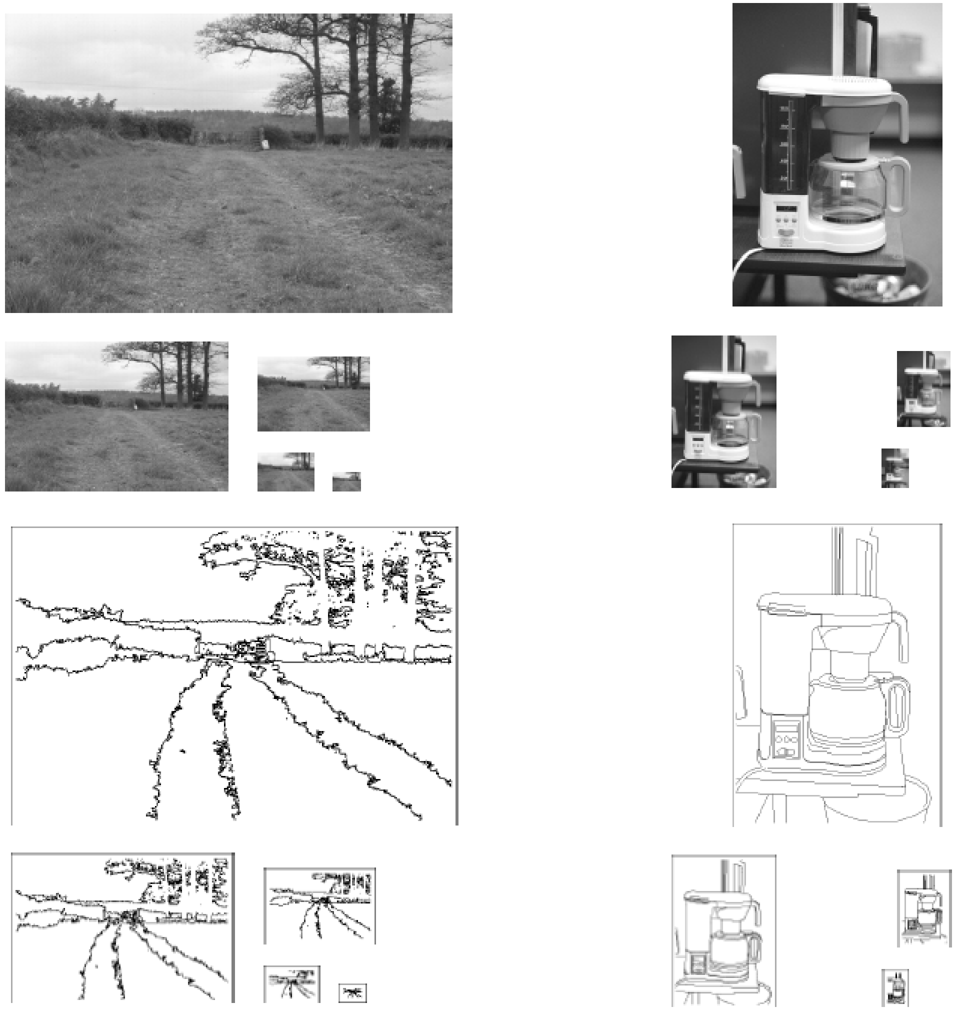
CompNets can localize attacks.



- C. Cosgrove et al. Robustness out of the box. Arxiv. 2020.
- A. Kortylewski et al. CISS. 2021.

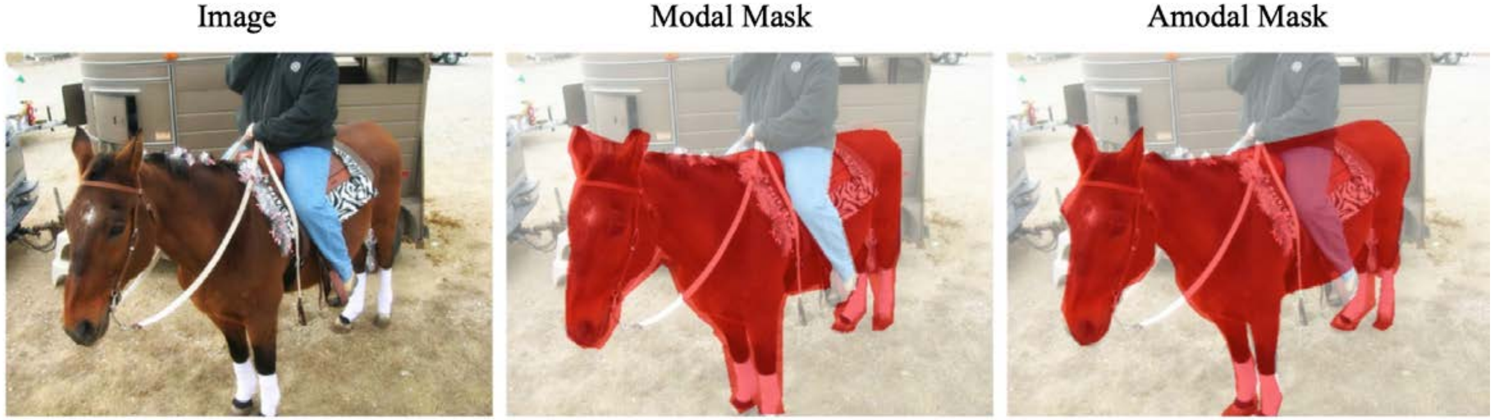
Edge Detection and Domain Transfer

- Bayesian/Generative methods enable domain transfer between two extremely different datasets.
- The Sowerby Dataset consists of outdoor images with much texture and vegetation.
- The South Florida dataset consist of indoor images with very little texture.
- But we can perform very well on either dataset using edge annotation only on the other.
- S. Konishi et al. TPAMI. 2003.



Foreground Background and Amodal Boundaries

- Segment objects into foreground and background (with additional annotation) and detect their amodal boundaries (under heavy occlusion).
- Example:



- Y. Sun et al. Arxiv. 2020.

References:

- C Cosgrove, A Kortylewski, C Yang, A Yuille. Robustness Out of the Box. arXiv. 2020.
- A. Kortylewski , J. He, Q. Liu, A. Yuille. Compositional Convolutional Nets, CVPR 2020.
- A. Kortylewski, Q. Liu, A. Wang, Y. Sun, A. Yuille. Compositional Convolutional Neural Networks. IJCV. 2020.
- A. Kortylewski, J. He, Q. Liu, C. Cosgrove, C. Yang, A. Yuille. Compositional Generative Networks and Robustness to Perceptible Image Changes. CISS. 2021.
- A. Wang, Y. Sun, A. Kortylewski, A. Yuille. Robust Object Detection under Occlusion with Context-Aware Compositional Nets. CVPR 2020.
- C. Yang, A. Kortylewski, C. Xie, Y. Cao, A. Yuille. Patch Attack. ECCV. 2020.
- A.L. Yuille & C. Liu. Deep Nets: What have they ever done for Vision. IJCV. 2021.

Discriminate Methods versus Generative Modeling

- For the last ten years most of computer vision research has been driven by discriminative methods (like Deep Networks). This has lead to huge improvements as evaluated by standard performance measures on large annotated datasets.
- But there are clear limits to the current approaches. When Deep Networks were introduced performance improved by 10AP for some visual tasks. But now improvements are much smaller (despite orders of magnitude more researchers and computer power).

Discriminate Methods versus Generative Modeling

- Fundamental Problem. The datasets are not nearly big enough. Too few visual tasks are being addressed (because of the need for annotation). Much tougher testing is required (e.g., out-of-distribution testing). The computer vision community is stuck in a local minimum.
- High Tech companies are aware of this problem. They have gigantic datasets (much bigger than the academic community) so they know that current methods are good, but not good enough, to deal with the complexity of the real world. "Existing methods do not work on big datasets because there are too many corner cases" (Anonymous CEO).

Discriminate Learning: The setup used by Deep Nets

- We first describe the standard procedures for learning a classifier assuming balanced annotated datasets.
- The standard procedure assumes there is an unknown distribution $P(x, y)$ which generates both the training and the testing data. Our task is to find a classifier $y = f(x : \theta)$ from the training data and with a loss function $L(y, f(x, \theta))$.
- We assume training samples $\mathcal{X}_{Train} = \{(x_i, y_i) : i = 1, \ldots, N\}$ and testing samples $\mathcal{X}_{Test}\{(x_a, y_a) : a = 1, \ldots, M\}$. Both are random samples from $P(x, y)$ so it assumed that $\mathcal{X}_{Train}$ and $\mathcal{X}_{Test}$ are similar.

Discriminate Learning: The setup used by Deep Nets

- The classifier is learnt from the training set to determine $\hat{\theta} = \arg\min_{\theta} \sum_{(x,y)\in\mathcal{X}_{Train}} L(y, f(x,\theta))$. The classifer is tested on the test set $\sum_{(x,y)\in\mathcal{X}_{Test}} L(y, f(x,\hat{\theta}))$.
- If there is sufficient training data, in terms of the complexity of the classifiers, then good performance on the training set will imply good performance on the testing set (must check to avoid overfitting).
- This is a *discriminative approach* because it learns a classifier $y = f(x:\theta)$. If the loss function is the cross entropy loss, then it tries to learn the distribution $P(y|x)$. *This standard approach can fail badly if we perform out-of-distribution testing*, i.e. if the test data is not generated from $P(x,y)$.

Why do People use the Discriminative Approach?

- *Why do people use the discriminative approach?*
- Two main reasons:
- (I) We know how to do discriminative learning. There is a long history of successful discriminative methods with Deep Nets being the most recent (and most effective). These methods are very successful for certain types of visual tasks (given data, GPUs, etc). They are essentially regression methods (using Statistics terminology, where regression includes continuous and discrete variables).
- (II) The current gold standard for evaluating computer vision (and other machine learning algorithms) is based on finite-sized balanced annotated datasets, like ImageNet and Coco. Papers are grant proposals are accepted based on algorithms performance on these types of datasets. Dsicriminative/regression methods are well suited to this type of task.

Why do People use the Discriminative Approach?

- There is an alternative approach – Generative/Bayesian – which formulates vision as *analysis by synthesis*. Synthesis means that the generative models are trained to generate images from the underlying visual scene. Analysis means that we interpret images by finding the visual scene which is most likely to generate the observed images. (A modern formulation is inverse computer graphics).
- But analysis by synthesis is much harder to do than discriminative models. There are good algorithms for generating images (Computer Graphics and Style-GANs), which are promising but not yet good enough. But, even more challengingly, we need algorithms which can invert the generative process to analyze the images.
- This talk will describe the advantages of the Generative/Bayesian approach compared to Discriminative methods. This will use approximate generative models. It can be thought of as approximate analysis by synthesis.

What are the limitations of the discriminative approach? (I)

- *What are the limitations of the discriminative approach?*
- The datasets are not big enough. And may never be big enough.
- Consider object classification on ImageNet, This is one of the big success stories of computer vision. On this dataset, computer vision algorithms seem to outperform human observers. But this is not true if you study performance closely.

What are the limitations of the discriminative approach? (I)

- The main limitation is that the datasets are not large enough to be representative of the complexity of the real world. It is easy to show that a sofa detector trained on ImageNet fails to classify sofa's which are seen from viewpoints unrepresented in the training data. There is also context bias (e.g., in ImageNet the only objects in trees are birds). There are *rare events*, also known as *corner cases*, which are underepresented in the dataset (both in the training and testing).
- This means that $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Test}}$ are *biased samples* from the real world distribution $P(x, y)$ so performance results on these datasets mail fail to generalize to other data that is sampled from $P(\vec{x}, y)$.

What are the limitations of the discriminative approach? (II)

- To make this more precise, suppose $y \in \mathcal{Y}$ (e.g., the set of object classes). For each object class y, we define $\mathcal{X}_y$ to be the set of all images in the real world that correspond to object class y. By contrast, our image examples in our training and testing datasets are $\mathcal{X}_{y,\mathrm{Train}}$ and $\mathcal{X}_{y,\mathrm{Test}}$. We assume that the training and testing datasets are balanced, which means that $\mathcal{X}_{y,\mathrm{Train}}$ and $\mathcal{X}_{y\mathrm{Test}}$ are similar (technically this means that these distributions are samples from the same distribution, but it may not be the real world distribution $P(x,y)$.).
- One simple type of Dataset bias arises if there is a large subregion $\mathcal{X}_{y_{\mathrm{bias}}} \in \mathcal{X}_y$ which does not overlap with $\mathcal{X}_{y,\mathrm{Train}}$ (and hence also from $\mathcal{X}_{y,\mathrm{Test}}$ because the training and test dataset is balanced). In this case, discriminative methods can give bad results if it is given images from $\mathcal{X}_{y_{\mathrm{bias}}}$. In practice, there are "missing subregions" for each object category, i.e. $\{\mathcal{X}_{y,\mathrm{bias}} : y \in \mathcal{Y}\}$.

What are the limitations of the discriminative approach? (II)

- This relates to the domain transfer problem. In this case, we have two different domains $\mathcal{X}_{1,\mathrm{Train}}, \mathcal{X}_{1,\mathrm{Test}}$ and $\mathcal{X}_{2,\mathrm{Train}}, \mathcal{X}_{2,\mathrm{Test}}$. But these are different so that algorithms trained on the first domain may not perform well on the second domain (and vice versa).
- Other biases can occur if the number of training examples $|\mathcal{X}_{y,\mathrm{train}}|$ differs between the object categories. They may be a lots of training data for some object classes y (i.e. $|\mathcal{X}_{y,\mathrm{train}}|$ is large) while for others classes the amount of training data is much less.

What are the limitations of the discriminative approach? (III)

- To get more intuitive understanding, realize that images are generated from the underlying three-dimensional world/environment. The image of an object is a function of different *environmental factors*: (i) geometric factors S, V (the shape of the object and the viewpoint), (ii) the T, M texture/material properties of the object, and (iii) the lighting L. (There are other factors, like occlusion or weather conditions, which we will discuss later). Formally $I = F(S, V, T, M, L)$.
- Dataset biases arise if objects are only seen from a limited range of these factors. E.g., the object is seen from a limited range of viewpoints, or a limited range of lighting conditions.

What are the limitations of the discriminative approach? (III)

- In ImageNet, objects are seen with background context. Ground truth for objects is specified by a bounding box surrounding the object which includes the foreground (the object) and the background context (the rest of the bounding box). Discriminative algorithms can exploit the background context (e.g. blue sky gives evidence that the object is an airplane or a bird).
- But this can lead to dataset biases which an algorithm can unfairly exploit. A deep net can incorrectly classify a penguin as a human if a TV is superimposed near the penguin (because a TV is background context that frequently occurs with humans but almost never with penguins).
- Note that these types of *environmental factors* are not modeled in deep networks. But they do appear in computer graphics models and some style-GANs models.

What are the limitations of the discriminative approach? (IV)

- We have illustrated the limitations for the task of object classification. But the same concerns arise for all visual tasks.
- The problems are also apparent for scene classification. Scenes consist of many objects arranged in backgrounds. The number of possibly ways to create scenes is combinatorial, which means that datasets which are representative must be truely gigantic.
- The problems are even clearer for action recognition. Studies suggest that many algorithms are largely relying on background context. E.g., "boxing" is classified by detecting the boxing ring, but boxers can box in the street (or in many locations) and, conversely, people in boxing rings may not be boxing (they could be playing poker as in the film "Lock, Stock, and Two Smoking Barrels"). In such situations, alternative measures like discriminating between very similar actions is a better performance measure.

What are the limitations of the discriminative approach? (IV)

- There are also many ways to defeat current deep networks by making small changes to images, which would not fool a human, but which cause deep networks to make serious mistakes.
- *More fundamental limitations of discriminative methods (at least of deep networks) is that they typically perform only a single task* (e.g., object detection, object classification, etc). By contrast, human can perform an enormous number of visual tasks (detect/classify objects, identify their parts and attributes, estimate their geometry and other environmental factors). Deep Nets do not represent these properties explicitly (if implicitly, then they are buried inside the features of the deep networks).

Out-of-Distribution

- An out-of-distribution task means that the algorithm is trained on date from distribution $P_1(\vec{x}, y)$ but tested on data from distribution $P_2(\vec{x}, y)$. There are other related tasks, such as deciding if a test image $\vec{x}$ is from $P_1(\vec{x}, y)$ or not.
- As stated, it is impossible to solve an out-of-distribution task without making additional assumptions.
- The simplest assumption is to find image features $\vec{f}(\vec{x})$ so that $P_1(y|\vec{f}(\vec{x})) \approx P_2(y|\vec{f}(\vec{x}))$ that they are likely to be invariant to details in the image which are invariant to the task. This has been successfully applied to some examples of domain adaption. But this is a very restrictive assumption.

Generative/Bayesian Perspective

- The Generative/Bayesian perspective is very different. It has, in theory, huge advantages compared to the discriminative approach. But it is harder and requires knowledge of generative/Bayesian methods which are not well-known to the computer vision community.
- A starting point for the Generative/Bayesian approach is the observation (a few slides ago) that image of object depend on environmental factors (shape/viewpoint, texture/material, lighting, etc). There is an external 3D world, the environment, which generate images which are inputs to AI vision systems (and the human visual system).
- This suggests that vision should be thought of as *modeling the environment*. This includes knowing that images consist of compositions of objects arranged on various background structure (e.g., roads, a grass lawn, a university lecture hall). Objects can be thought of as compositions of elementary parts (a horse consists of a head, torso, legs, and tail) which, in turn, can be expressed in terms of subparts (e.g., head contains eyes, nose, mouth, etc.). Actions can be thought of as actors (objects) interacting with each other obeying spatiotemporal relationships).

Generative/Bayesian Perspective

- This perspective suggests that all the variables (environmental factors, etc.) should be represented explicitly. This has many advantages, it enables us to build new objects from existing parts (and recognize that a banana consists of banana slices enveloped in a skin). It also allows us to perform multiple tasks in a consistent manner (their answers will be given by these internal representations. (Question: can we design a taxonomy of out-of-distribution tasks?). (Approximate analysis by synthesis – use features that are invariant to factors/details which we do not care about).
- Note: generalizing to unseen colors/textures/geometries requires the concepts of 3D models and the factorization of images in terms of geometry/viewpoint, texture/material, and lighting/illumination.

Toy Example: Contaminated Data Samples

- We start by describing a toy example of out-of-distribution testing and a Bayesian approach for dealing with it. We assume that some of the testing data is generated from the training distribution $P(\vec{x}, y)$ and the rest is generated by another distribution $Q(\vec{x}, y)$. This can be done by introducing a latent variable $z \in \{0, 1\}$ for each data sample, where $z = 1$ if the data is generated by $P(\vec{x}, y)$ and $z = 0$ if it is generated by $q(\vec{x}, y)$.
- So the test data is generated by $Pr((\vec{x}, y)|z)P(z)$ where $Pr((\vec{x}, y)|z) = \{P(\vec{x}, y\}^z \{Q(\vec{x}, y)\}^{1-z}$ and $P(z = 1) = 1 - \epsilon$ $P(z = 0) = \epsilon$. In other words, the test data is generated from $(1 - \epsilon)P(\vec{x}, y) + \epsilon Q(\vec{x}, y)$, where $Q(\vec{x}, y)$ is another distribution and ϵ is a constant.

Toy Example: Contaminated Data Samples

- If ϵ is small, then classifiers trained on data from $P(\vec{x}, y)$ may still perform well on data from $(1-\epsilon)P(\vec{x}, y) + \epsilon Q(\vec{x}, y)$. But performance will typically degrade badly if ϵ is large. We can also estimate the latent variable z to determine if the data is likely to come from $P(\vec{x}, y)$ or $Q(\vec{x}, y)$.
- Note: the discipline of Robust Statistics partially addresses this problem by showing that some distributions $P(\vec{x}, y)$ are *robust* in the sense that they are unaffected by small amounts of contamination (small ϵ). Gaussian distributions, for example, are known to be non-robust.

Toy Example: Contaminated Data Samples

- How to address this problem? One solution is to use Bayesian methods. Instead of learning the classifiers we learn generative probability distributions $P(\vec{x}|y)$ and $P(y)$ from the training set. We estimate $Q(\vec{x}, y) = Q(\vec{x}|y)Q(y)$ using other data (or modelling assumptions).
- Suppose the data is generated by $Pr((\vec{x}, y)|z)P(z)$. We can estimate z to determine of the data comes from $P(\vec{x}, y)$ or $Q(\vec{x}, y)$ (e.g., by comparing $\max_y Q(y|\vec{x})$ with $\max_y P(y|\vec{x})$ using a weighted threshold to allow for ϵ and model complexity). And then estimate $\hat{y}$ from $P(y|\vec{x})$ or $Q(y|\vec{x})$ as appropriate.

Toy Example: Contaminated Data Samples

- This is the Bayesian/Generative approach. It is conceptually simple but it requires us to learn probability distributions $P(\vec{x}, y)$ and $Q(\vec{x}, y)$ for generating images. But it is very difficult to learn probability distributions for images because the dimensionality of images is very high. It is much easier to learn discriminative distributions $P(y|x)$ because these are much lower-dimensional.
- Recently there is hope that we can make generative models of objects, and perhaps even scenes, using a combination of computer graphics, GANs, and other techniques.

Domain Generalization: Edge Detection Example

- We now consider domain generalization, which is a variant of out-of-distribution learning. This is a based on Konishi *et al.* "Statistical Edge Detection". TPAMI. 2003.
- We assume two different edge detection domains (the Sowerby and the South Florida datasets). These are specified by $P_1(\vec{x}, \vec{y})$ and $P_2(\vec{x}, \vec{y})$. These distributions are very different (Sowerby and South Florida contain outdoor and indoor images respectively).

Domain Generalization: Edge Detection Example

- Here $\vec{x} = \{x_a : a \in \mathcal{D}\}$ are image feature vectors (e.g., image derivatives, which are cues for edges) defined at positions a in the image lattice $\mathcal{D}$ and the variables $y_a \in \{0, 1\}$ denote whether there is an edge at a ($y_a = 1$) or not ($y_a = 0$).
- We re-express the distributions as $P_1(\vec{x}, \vec{y}) = P_1(\vec{x}|\vec{y})P_1(\vec{y})$ and $P_2(\vec{x}, \vec{y}) = P_2(\vec{x}|\vec{y})P_2(\vec{y})$. To simplify we assume that the distributions are factorizable, i.e. $P_i(\vec{x}|\vec{y}) = \prod_{a\in\mathcal{D}} P_i(x_a|y_a) \text{ for } i = 1, 2$. Similarly for the distributions $P_1(\vec{y}), P_2(\vec{y})$.
- If we know these distributions, then we can detect edges in the two domains by thresholding $\log \frac{P_1(x_a|y_a=1)}{P_1(x_a|y_a=0)}$ and $\log \frac{P_2(x_a|y_a=1)}{P_2(x_a|y_a=0)}$ respectively. This was the first effective statistical/probabilistic edge detector (Konishi *et al.* 2003) which even outperformed s discriminative method (Martin *et al.* 2004).

Domain Generalization: Edge Detection Example

- We now consider domain adaptation. We have annotated training data from the first domain, i.e., we can estimate $P_1(\vec{x}, y), P_1(\vec{x}|y), P_1(y)$, and we have unannotated data from the second domain, i.e., we know $P_2(\vec{x})$. If the domains are sufficiently different, i.e. $P_1(y|\vec{x}) \neq P_2(y|\vec{x})$, then a classifier trained on the first domain will not perform well on the second domain. (Edge detectors trained on South Florida perform badly when tested on Sowerby).
- For the second dataset, *we assume that the distributions of the feature vectors are similar at the edges*, i.e. $P_1(x_a|y_a = 1) = P_2(x_a|y_a = 1)$. This is reasonable because at the edges there is typically a big discontinuity in the image intensity (and can be relaxed). Hence we can use this to estimate $P_2(x_a|y_a = 1)$. Similarly we assume $P_1(\vec{y}) = P_2(\vec{y})$.

Domain Generalization: Edge Detection Example

- To estimate $P_2(x_a|y_a = 0)$, we exploit the fact that *most pixels in images are not edges* so we can approximate it, without needing annotations, by $P_2(x_a) = P_2(x_a|y_a = 0)P_2(y_a = 0) + P_2(x_a|y_a = 0)P_2(y_a = 1)$.
- In practice, Konishi *et al.* relaxed the assumption that $P_1(x_a|y_a = 1) = P_2(x_a|y_a = 1)$, and assumed instead that the magnitude of the filter responses between the two datasets differed by an unknown constant scaling factor which could be estimated by making assumptions about the form of the probability distributions.

Detecting/Classifying Objects under occlusion

- Now consider a model for classifying input $\vec{x}$ as an object y if there is occlusion. We describe a simplified version of CompNeta (A. Kortylewski et al. CVPR 2020).
- The input $\vec{x}$ is $\{x_a : a \in \mathcal{D}\}$ where a labels position within the spatial domain $\mathcal{D}$. We assume that the training data is generated by a distribution $P(\vec{x}, y)$ and consists of objects which are mostly unoccluded.
- A discriminative approach (e.g., standard Deep Net) would learn a classifier like $P(y|\vec{x})$ (plus a threshold). But performance of this classifier will start degrading if it is tested on objects which are occluded.

Detecting/Classifying Objects under occlusion

- *The default discriminative strategy* is to create a bigger dataset with occluded objects, but this is only partly successful. This is arguably because the number of ways you can occlude an object are combinatorial so the training dataset may not big enough.
- Instead we consider an alternative approach using bayesian/generative methods. From the training data, drawn from $P(\vec{x}, y)$ we learn/estimate. $P(\vec{x}|y)$ and $P(y)$. For simplicity, we assume that the objects are only seen from a fixed viewpoint. The CompNets were more complicated because they had other latent variables representing parts(see later).

Detecting/Classifying Objects under occlusion

- For these bayesian/generative models $P(\vec{x}|y)$ and $P(y)$, the feature vectors $\vec{x}$ were chosen to be features filters at the top convolutional layer of a deep network.
- This is because convolutional deep network features are *invariant to the details of the objects* (lighting, texture, etc.) which are unimportant for detection/classification (intuitively, this follows from how standard Deep Nets are trained) This means we can use simple generative/bayesian models which are easy to learn and to do inference (i.e. detect and classify objects), enabling us to perform approximate analysis by synthesis.
- In particular we assume that the distributions are factorizable, i.e. $P(\vec{x}|y) = \prod_{a \in \mathcal{D}} P(x_a|y)$. (But this can be relaxed theoretically. The key property is that we need to be able to compute the marginal distribution for a subset of the image pixels $\{a \in \mathcal{D}\}$, which is easy if the model is factorizable). $P(x_a|y)$ is a parameterized distribution (see later for the form of the distribution).

Detecting/Classifying Objects under occlusion

- To introduce occlusion we assume that some of the components of the input $\vec{x}$ are generated by another distribution $Q(\vec{x})$, which we also assume is factorizable $Q(\vec{x}) = \prod_{a \in \mathcal{D}} Q(x_a)$ (and independent of position).
- This distribution $Q(\vec{x})$ is learnt separately from $P(\vec{x}, y)$. $Q(\vec{x})$ can be learnt by selecting image regions which do not include the objects. (which requires no annotation).

Detecting/Classifying Objects under occlusion

- We introduce a latent variable $\vec{z} = \{z_a : a \in \mathcal{D}\}$, with $z_a \in \{0, 1\}$ where $z_a = 1$ means the data at position a is generated by an object and $z_a = 0$ means it is generated by an occluder.
- The variables $\vec{z}$ define two distinct, and complimentary, subregions of $\mathcal{D}$. $\mathcal{D}(\vec{z}) = \{a : z_a = 1\}$ and $\bar{\mathcal{D}}(\vec{z}) = \{a : z_a = 0\}$. Hence $\mathcal{D}(\vec{z}) \bigcup \bar{\mathcal{D}}(\vec{z}) = \mathcal{D}$ and $\mathcal{D}(\vec{z}) \bigcap \bar{\mathcal{D}}(\vec{z}) = \phi$.

Detecting/Classifying Objects under occlusion

- Now we construct a distribution for generating data with occlusion. This is of form $Pr(\vec{x}|y,\vec{z}) = P(\{x_a : a \in \mathcal{D}(\vec{z})\})Q(\{x_a : a \in \bar{\mathcal{D}}(\vec{z})\})$ with a distribution $P(\vec{z})$ (the probability of occluded data). Note that the form of $Pr(\vec{x}|y,\vec{z})$ requires us to compute the marginals of $P(\vec{x}|y)$ and $Q(\vec{x})$ (easy if both distributions are factorizable as they are in the CompNet papers).
- The out-of-distribution challenge is how to estimate the object y and the latent occluder variable $\vec{z}$ for data generated by $Pr(\vec{x}|y,\vec{z}), P(\vec{z})$, if we have only trained on data from $P(\vec{x},y)$ and can learn $Q(\vec{x})$ independently.

Detecting/Classifying Objects under occlusion

- The Bayesian approach gives a principled way to address this problem.
- (I) Learn $P(\vec{x}|y)$ from the training data.
- (II) Learn $Q(\vec{x})$ from other (unannotated data).
- (III) Estimate $\vec{z}$ and y by maximizing $Pr(\vec{x}|y, \vec{z})P(y)$ with respect to $\vec{z}$ and y. For these types of models computing these quantities is straightforward (similar to the feedforward pass of a deep net).

Detecting/Classifying Objects under occlusion

- How important is it to learn the distributions $P(\vec{x}, y)$ and $Q(\vec{x})$ precisely? This is perhaps best studied empirically because theoretical studies are hard to do. But there are some underlying concepts.
- The first is the notion of robustness (see literature on Robust Statistics). Distributions are non-robust if small changes to them could dramatically influence the estimates (the Gaussian is a classic example of a non-robust distribution). There are also attempts to see under what situations approximating distributions affects performance (e.g., A. Yuille et al. "Order parameters: when does high level knowledge help" CVPR 1999). But these types of theoretical studies are hard to do and their results are not of great interest to researchers in computer vision.

Detecting/Classifying Objects under occlusion

- *The value of the Bayesian/generative formulations is they enable us to break these problems down into different components, e.g., $P(\vec{x}, y)$ and $Q(\vec{x})$, which can be estimated separately*.
- Perhaps discriminative methods could be adapted to these tasks given enough ingenuity (and perhaps be fine-tuning with small validation sets). Alternatively, it might be possible to generate an enormous dataset to estimate $Pr(\vec{x}|y, \vec{z})$. But this would probably need to so big that it would be more efficient to exploit the fact that $P(\vec{x}, y)$ and $Q(\vec{x})$ can be learnt independently.
- This can, of course, be generalized to much more complicated situations.

Detecting/Classifying and Segmenting Objects under occlusion

- We now want to learn a model which is more complex in three respects: (I) The model can can perform segmentation, as well as classification and detection, and can also deal with different viewpoints. (II) The model contains latent variables which must be estimated during learning. (III) The model can generalized out-of-distribution to be robust to occluders and also to localize them.
- The model is evaluated both for the tasks of object classification and *amodal completion*, which is to detect the object boundaries which are invisible (because they are occluded). Amodal completion get increasingly difficult when the amount of occlusion is increased, so it is well suited to our approach. (Note: humans are good at this task but AI algorithms are not.) Note that we train the algorithm for one task (classification) but we also evaluate it for another (amodal completion).

Detecting/Classifying and Segmenting Objects under occlusion

- We learn/train the model on data which is not occluded but where the only annotation is the class label of the object. This means that the mixture/viewpoint and the boundary are latent variables which must be estimated during learning (the EM algorithm will be used).
- Then we generalize out-of-distribution to the case where there are occlusions. The final model will be able to classify the object, detect its boundaries and in particular its amodal boundaries, and also detect the occluders.

Detecting/Classifying and Segmenting Objects under occlusion

- We first address the problem of formulating the model and learning it from training data without occlusions. They we will generalize the model out-of-distribution so that it will work on objects which are heavily occluded.
- We assume that each training bounding box $\mathcal{D}$ contains a foreground region for the object and a background context region. We define a latent variable $\vec{w} = \{w_a\}$, where $w_a = 1$ if pixel a is part of the foreground object and $w_a = 0$ if it is part of the background.
- We also define another latent variable c which indicates the center of the object (we assume that the center of the object is displaced from the center of $\mathcal{D}$ by c).

Detecting/Classifying and Segmenting Objects under occlusion

- We introduce a latent variable m for each object which will (roughly) correspond to the viewpoint of the object. This is needed because we will learn a generative model for the feature vectors of the object and this generative model will depend on the viewpoint. There are also other latent variables for object parts, but we will not discuss them because of space limitations.
- We slightly change notation so that the feature vectors are given by $F = \{f_a : a \in \mathcal{D}\}$. As before, these are convolutional features at the top level of a deep net (which are invariant to unimportant details of the objects).

Detecting/Classifying and Segmenting Objects under occlusion

- The generative model is defined for the foreground and background context of the object. The background context is the local region surrounding the object (i.e. the region within the bounding boxe which is not the object).
- Background context can be a useful cue for detecting and classifying objects (e.g., the background context of an airplane is usually sky) and standard deep networks exploit it. But context can be missleading (changing the background context can cause deep nets to make mistakes). Our approach, where foreground and background are distinguished by a latent variable makes it possible to use the context when it is helpful and ignore it when it is not.

Detecting/Classifying and Segmenting Objects under occlusion

- We define the generative model to be a generative model of the feature vectors F conditioned on the object y, the mixture component m, and the foreground/background $\{w_a : a \in \mathcal{D}\}$: $P(F|y, m, \vec{w}) = \prod_{a \in \mathcal{D}} P_a(f_a|y, m)^{w_a} B_a(f_a|y, m)^{1-w_a}$ and $P(\vec{w}|y, m) = \prod_{a \in \mathcal{D}} P_a(w_a|y, m)$. We can assume that $P(m|y)P(y)$ are known.
- The distributions $P_a(f_a|y, m)$ and $B_a(f_a|y, m)$ are expressed as mixtures of von Mises Fisher distributions: $P(f_a|\mathcal{A}_a^{y,m}, \Lambda) = \sum_k \alpha_{i,k}^{y,m} P(f_a|\lambda_k)$ and $B(f_a|\mathcal{A}_a^{y,m}, \Lambda) = \sum_k \zeta_{i,k}^{y,m} P(f_a|\lambda_k)$, where $p(f_a|\lambda_k) = \frac{e^{\sigma_k \mu_k^T f_a}}{Z(\sigma_k)}, ||f_a|| = 1, ||\mu_k|| = 1$ and $Z(\sigma_k)$ is a normalization constant. The parameters $\{\alpha_{i,k}^{y,m}\}$ and $\{\zeta_{i,k}^{y,m}\}$ are learnt.

Detecting/Classifying and Segmenting Objects under occlusion

- We can learn the model using the EM algorithm to deal with the latent variables. This can be done by a combination of differentiation (to estimate parameters like $\{\alpha^{y,m}_{i,k}\}$ and $\{\zeta^{y,m}_{i,k}\}$ and clustering algorithms to estimate the mixture variables, the parameters $\{\lambda_k\}$ of the von Mises Fisher mixtures and the foregroun/background variables $\{w_a\}$, and the distributions $P(w_a|y,m)$.
- It can be shown that the mixture variables roughly correspond to different viewpoints of the objects, the parameters $\{\lambda_k\}$ specify a dictionary of object parts, the variables $\{w_a\}$ correspond to a foreground/background segmentation of the object, and the distributions $P(w_a|y,m)$ correspond to shape masks of the objects for different viewpoints.

Detecting/Classifying and Segmenting Objects under occlusion

- Like all EM algorithms, the success depends on good initialization. E.g., to estimate the mixtures we use spectral clustering to group stimuli into classes with similar appearance.to estimate the foreground/background variables we can use log-likelihood tests between $P(f_a|\mathcal{A}_a^{y,m}, \Lambda)$ and $B(f_a|\mathcal{A}_a^{y,m}, \Lambda)$ before the $P(w_a|y, m)$ are estimated. WE can also impose a prior $P(\{w_a\})$ on the foreground/background variables to encourage neighboring pixels to either be foreground or background.
- The learning is formulated as maximum likelihood (or maximum a posteriori if we use priors). We can also add regularizing terms. We can also either use deep network features which are pre-trained (e.g., on ImageNet) or where these features are also trained. Formally the learning can look like end-to-end deep network training but using a loss function which differs from standard deep networks. (Calling it end-to-end training makes reviewers happy!).

Detecting/Classifying and Segmenting Objects under occlusion

- To generalize this model to work out of distribution we add an occlusion process as for our earlier model. we introduce binary valued occlusion variables $\{z_a\}$ where $z_a \in \{0,1\}$ indicates whether pixel a is occluded or not occluded ($z_a = 0$ or 1 respectively).
- The occluder distribution $Q(F) = \prod_{a \in \mathcal{D}} Q(f_a)$ where $Q(f_a)$ is a mixture of von Mises Fisher distributions $Q(f_a) = \sum_k q_k p(f_a|\lambda_k)$, with $p(f_a|\lambda_k) = \frac{e^{\sigma_k \mu_k^T f_a}}{Z(\sigma_k)}, ||f_a|| = 1, ||\mu_k|| = 1$ and $Z(\sigma_k)$ is a normalization constant. This can be learnt from background data. Note this distributon differs from the background context distribution (which depends on the spatial position and learns the typical context of the objects). We also specify, but do not learn, a prior $P(\vec{z}) = \prod_{a \in \mathcal{D}} P(z_a)$ where $P(z=0)$ is a rough measure of how much occlusion we want the algorithm to be able to deal with. We also introduce a displacement variable c (cite Wang).

Detecting/Classifying and Segmenting Objects under occlusion

- This gives a model of form:

$$\begin{aligned} P(F|y,m,\vec{w},\vec{z},c) &= \prod_{a\in\mathcal{D}} P_{a-c}(f_a|y,m)^{w_a z_a} \\ &\times B_{a-c}(f_a|y,m)^{(1-w_a)z_a} Q(f_a)^{(1-z_a)}. \end{aligned} \tag{1}$$

$$P(\vec{w}|y,m,c) = \prod_{a\in\mathcal{D}} P_{a-c}(w_a|y,m) \tag{2}$$

Detecting/Classifying and Segmenting Objects under occlusion

- From these distributions we can estimate the object class (and the class mixture), the center c of the object, the occlusion map $\{z_a\}$, and the foreground background map $\{w_a\}$.
- For the later two, we also use the priors $P(\vec{w}|y,c)$ and $P(\vec{z})$. Following standard Bayesian procedures, we will estimate distributions for all the quantities and then threshold them if we want to make a decision.
- From our estimates of foreground/background and the occluders we can determine the amodal boundaries by finding the boundaries between the occluded foreground regions (i.e. $w_a = 1, z_a = 0$) and the background context regions (i.e. $w_a = 1$).

Detecting/Classifying and Segmenting Objects under occlusion

- For Amodal completion the shape prior $P(\{w_a\}|y, m)$ is very useful because it gives prior knowledge of the likely boundary of the object in the occluded regions (which, of course, requires that the model is able to classify the object and estimate its mixture component). The visible boundary of the object can be detected by combining local evidence for the foreground and backgroun – from $P(.)$ and $B(.)$ – with the shape prior.
- The most successful algorithms for Amodal completion require annotations for the segmentation including the invisible regions. These outperform our algorithm if the leverl of occlusion is low. But we outperform them if the occlusion level is high. We outperform all other alternative algorithms. In addition, we observe that introducing the shape prior improve performance of the CompNet for classification (compared to CompNets without shape priors.

Detecting/Classifying and Segmenting Objects under occlusion

- These notes have illustrated how some out-of-distribution tasks can be formulated in terms of Bayesian/generative probability models. This enables us to take models which have been trained on data from one distribution and modify them in a principalled manner so that they work on data from other distributions. We have concentrated mostly on generalizing out-of-distribution to deal with occlusion, but we also discussed domain transfer.
- One conclusion is that Bayesian/Generative methods are useful for formulating out-of-distribution tasks. They can also be used for learning latent variables (e.f.,foreground/background) which enables us to train the model for one task but also enable the model to perform other tasks in a consistent manner. They also yield a strategy to ensure that models are interpretable, by validating the latent variables against some ground truth (e.g., we know we have detected a car because we can find wheels at position xxx, the door at position yyy, and so on).

Detecting/Classifying and Segmenting Objects under occlusion

- Most out-of-distribution tasks can be formulated in this manner. Obvious examples include recognising objects from novel viewpoints, or under different lighting, or with different texture/materials. Some of these need 3D models of objects, which might be learnt from Computer Graphics, style-GANs, compositional models, or some combination. To use these models requires inverse inference which is challenging. CompNets simplies this by doing approximate analysis by synthesis where we generate feature vectors which are invariant to details that are unimportant for the task.

Conclusion

- The Bayesian/Generative approach has many advantages compared to the discriminative approach.
- But its strengths are not apparent on standard performance measures (balanced annotated datasets).Its strengths become clearer on out-of-distribution.
- It does require more work than discriminative models. We would need to construct generative models of all objects and scenes.