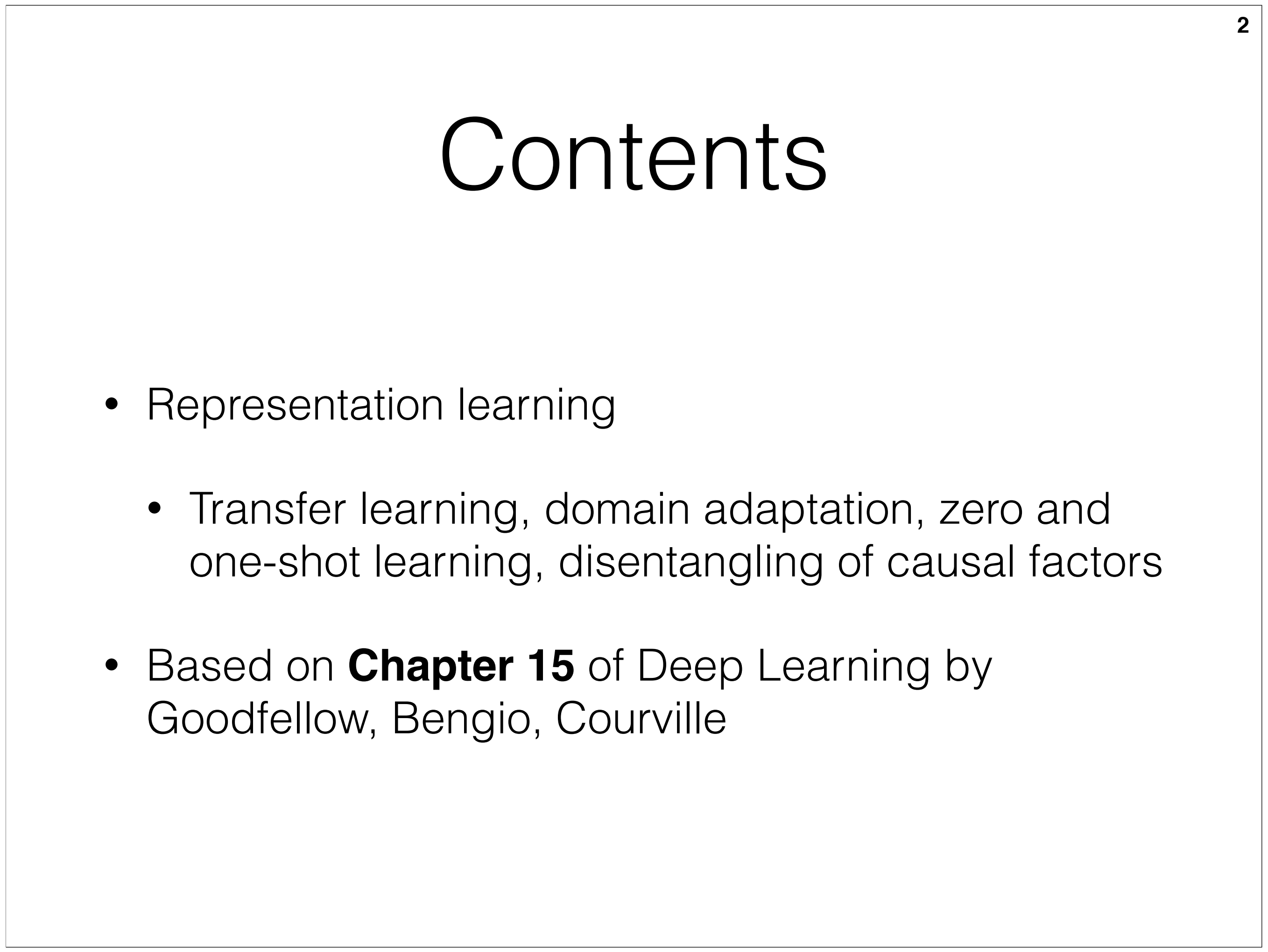
u^b

b
UNIVERSITÄT
BERN

Representation Learning

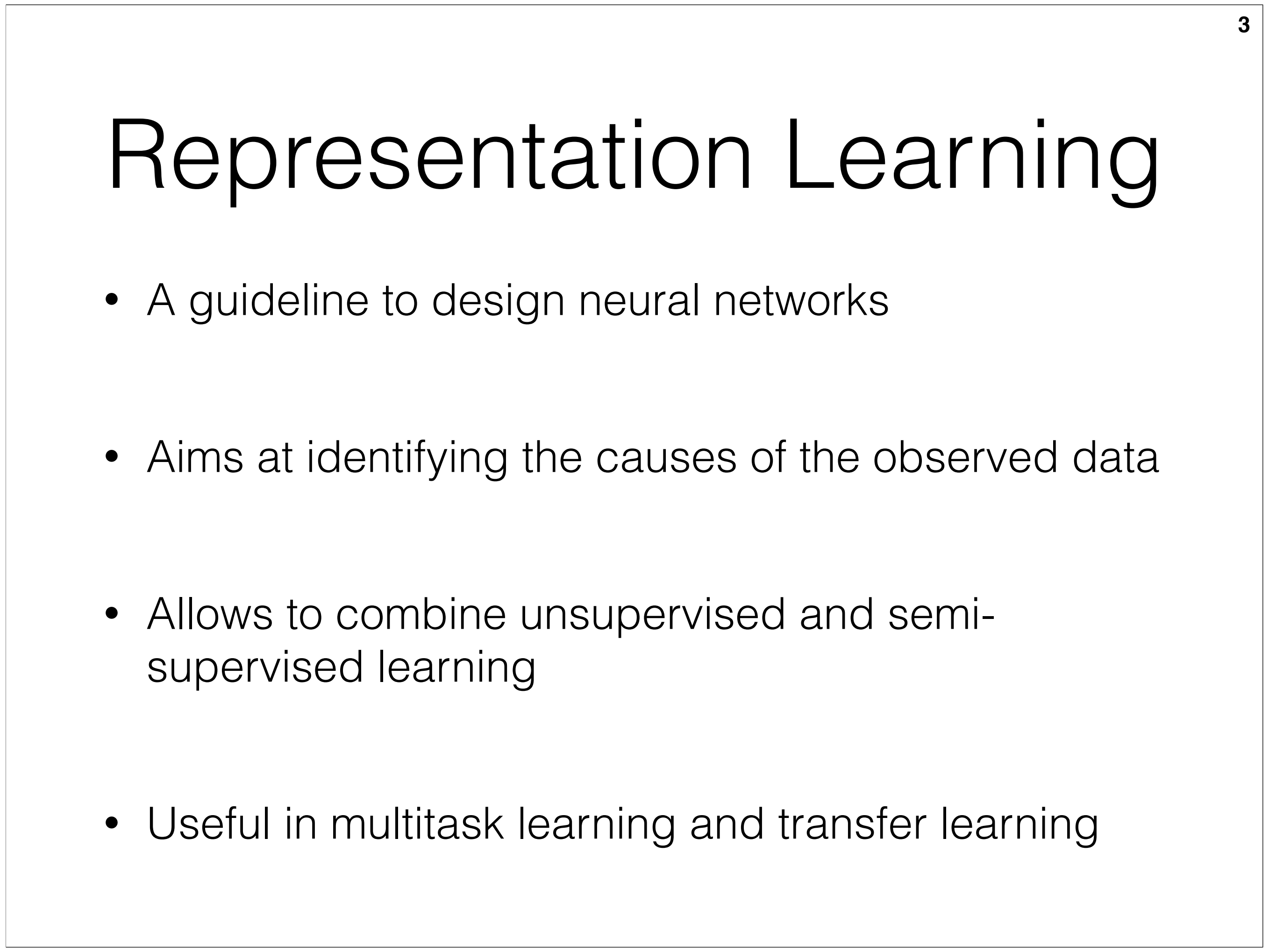
Paolo Favaro



Contents

- Representation learning
  - Transfer learning, domain adaptation, zero and one-shot learning, disentangling of causal factors
- Based on **Chapter 15** of Deep Learning by Goodfellow, Bengio, Courville

We start by defining representation learning and seeing in what areas it applies. Then we go deeper into what a good representation is and its relation to causal factors of variation.
Finally we discuss the model for a representation. In particular we compare a local with a distributed representation.



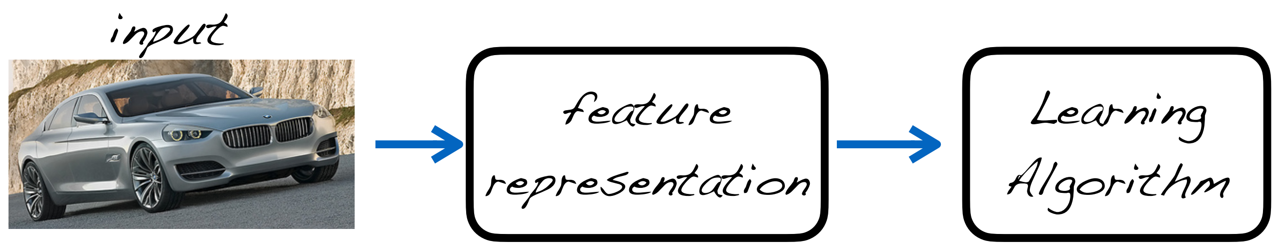
Representation Learning

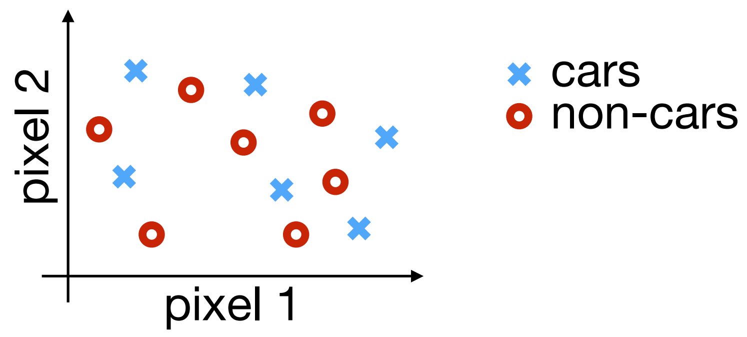
- A guideline to design neural networks
- Aims at identifying the causes of the observed data
- Allows to combine unsupervised and semi-supervised learning
- Useful in multitask learning and transfer learning

What is representation learning?
It turns out that a useful representation finds out the causal factors that explain data.

Where can it be used?
It has applications in unsupervised, semi-supervised, multitask and transfer learning.
We will see these applications in the next slides

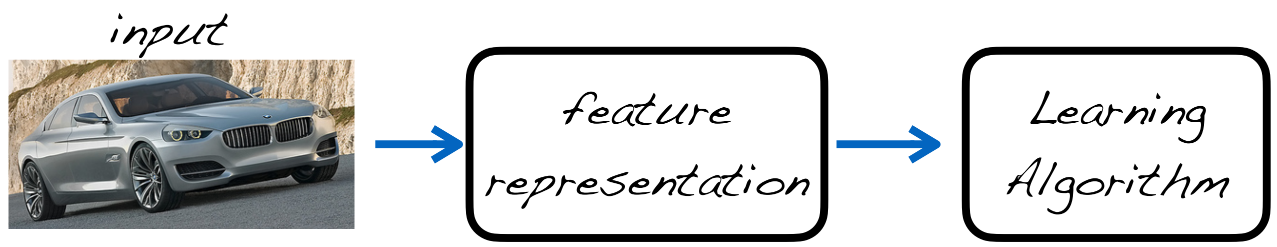
Representation Learning

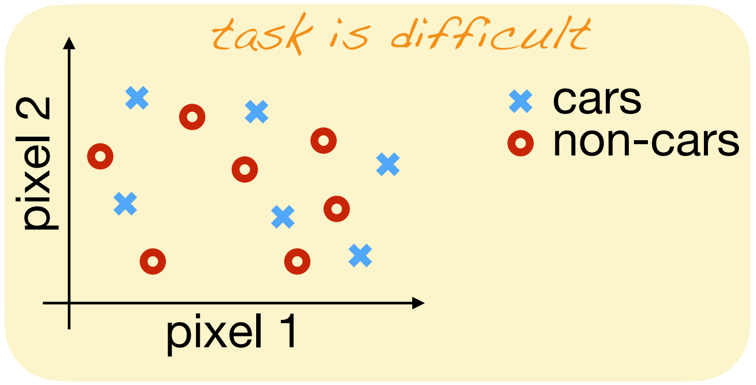




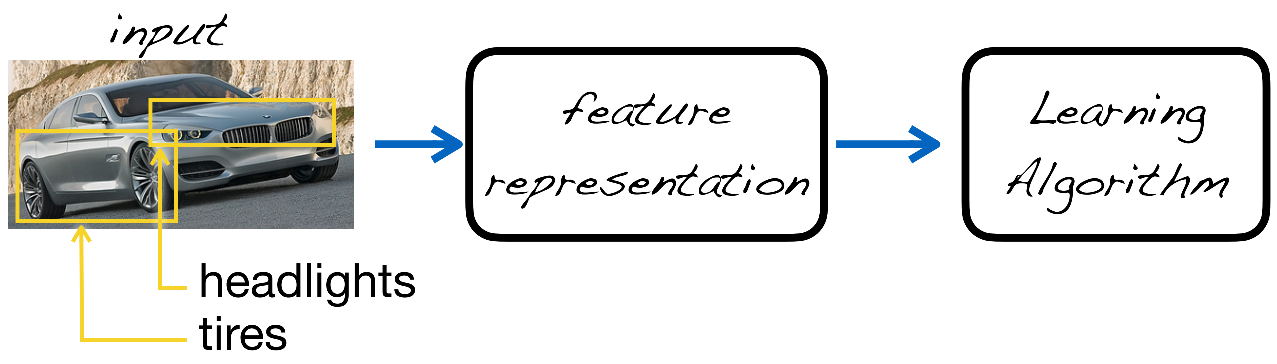
The complexity of a task depends on how information is presented.
For example, in the case of the classification of an image as containing a car or not, when the information is presented as an array of pixel values the task is extremely difficult. When instead the information is presented with an appropriate feature representation (e.g., tires and headlights), the task becomes very easy.

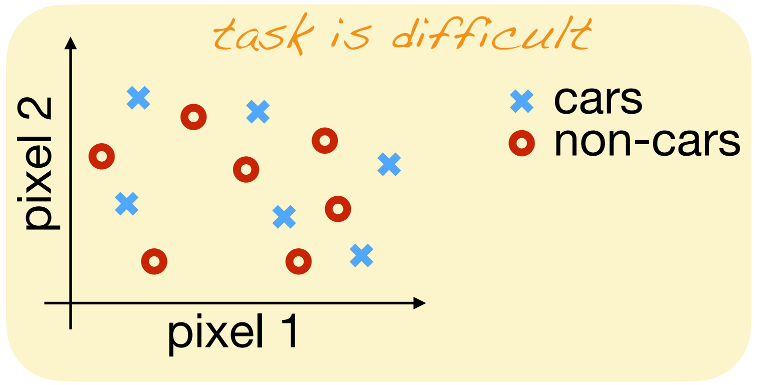
Representation Learning



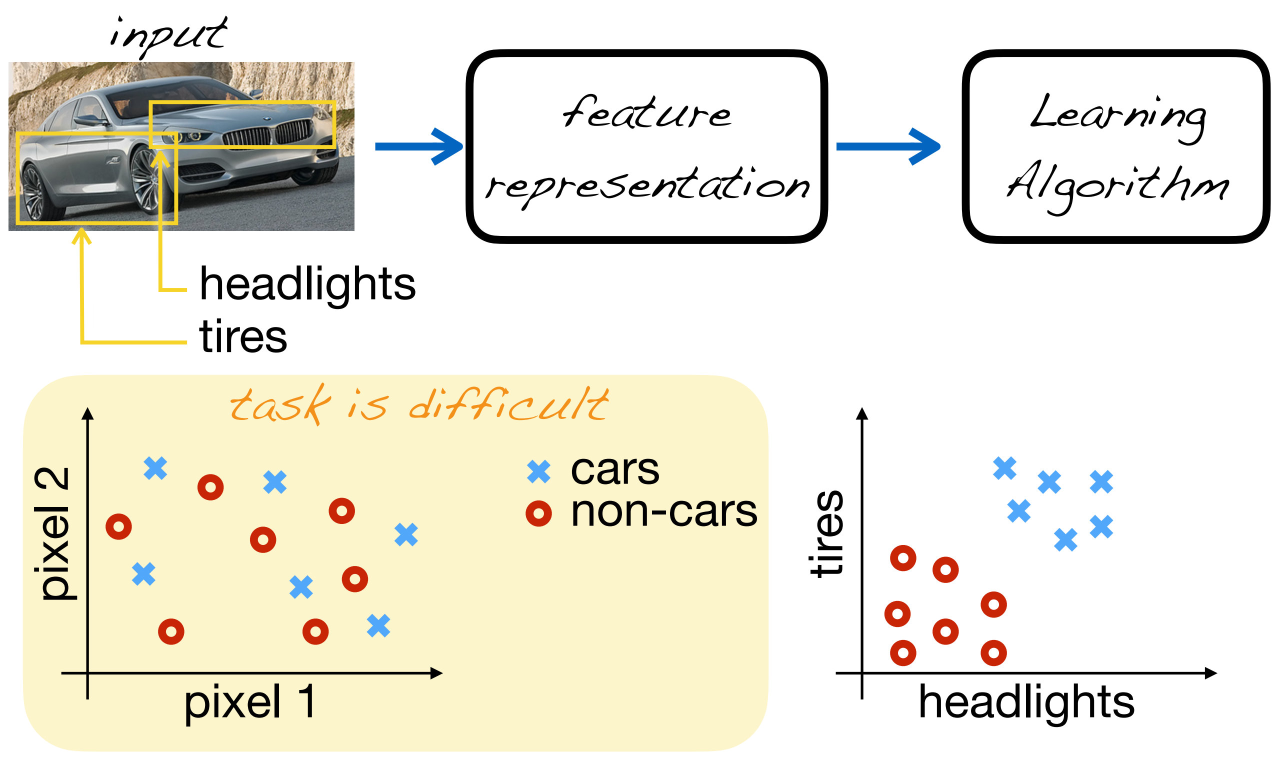


Representation Learning

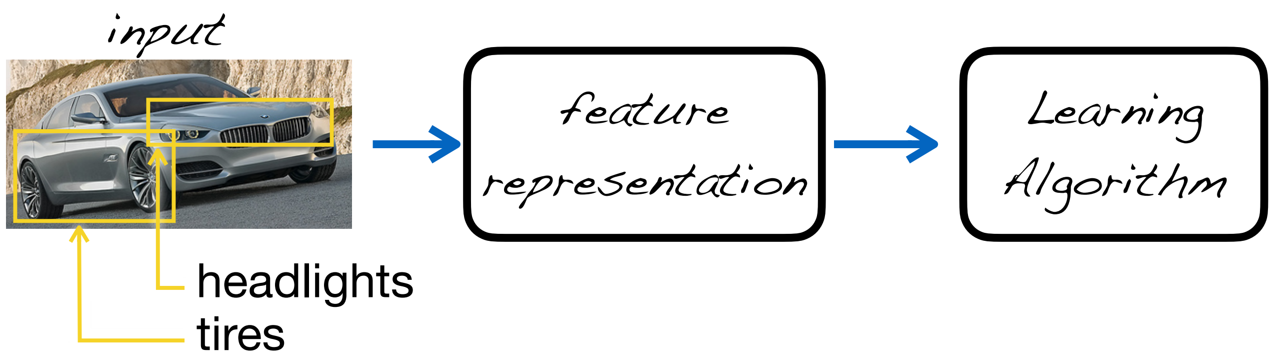


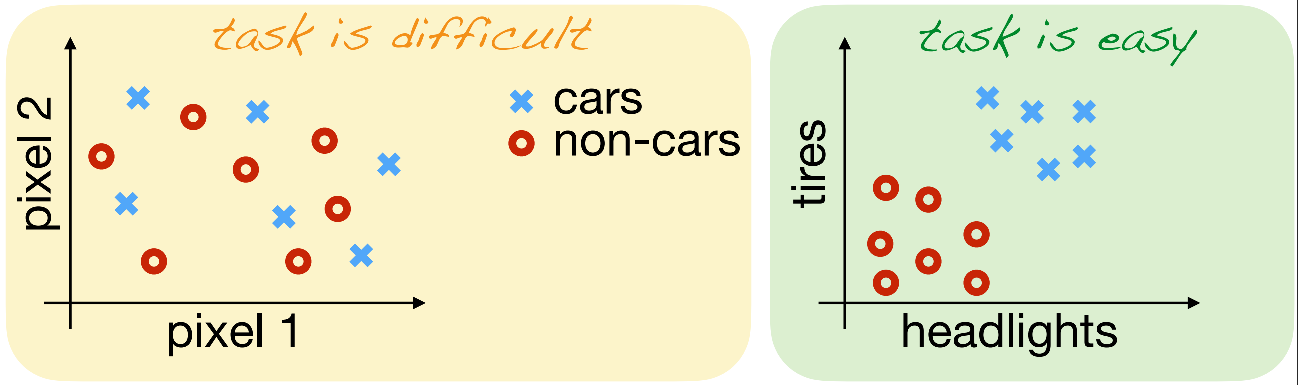


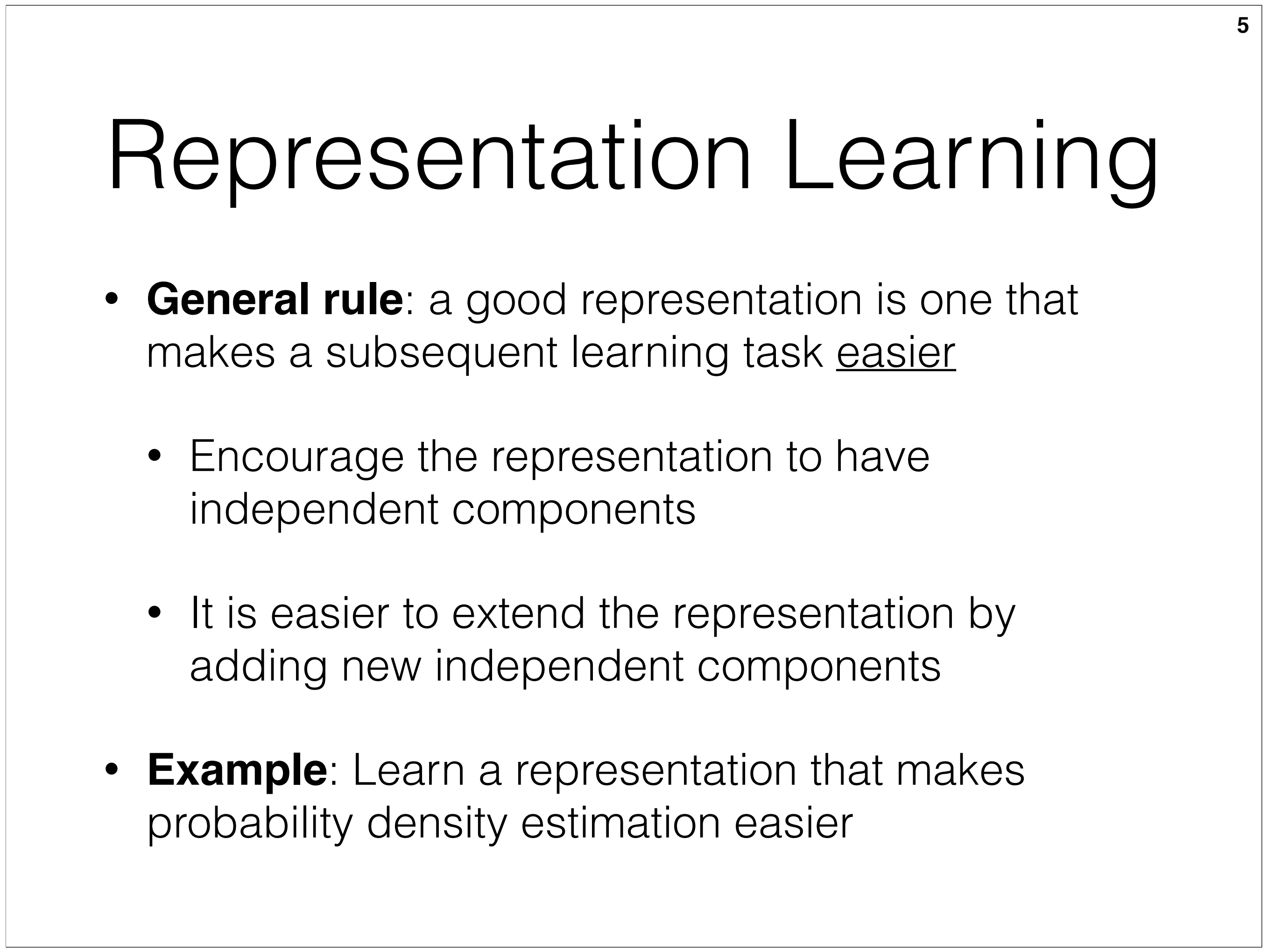
Representation Learning



Representation Learning







Representation Learning

- **General rule**: a good representation is one that makes a subsequent learning task <u>easier</u>
  - Encourage the representation to have independent components
  - It is easier to extend the representation by adding new independent components
- **Example**: Learn a representation that makes probability density estimation easier

About the general rule: In this way representations are task-dependent.

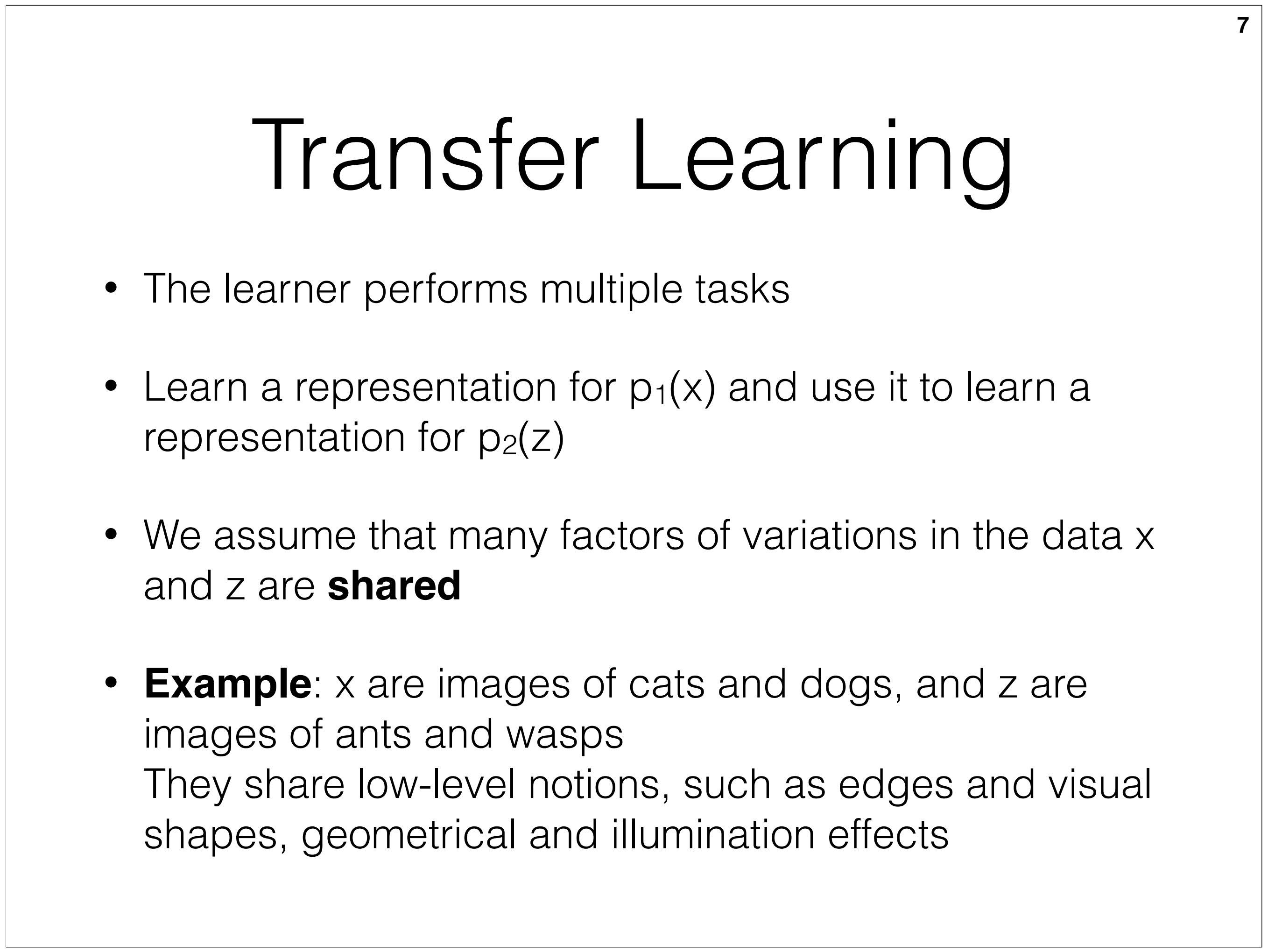
About the example: Distributions with independency are easier to model.

Let us now see how representation learning applies to several learning domains.

Applications

- Transfer learning
- Domain adaptation
- One/Zero-shot learning
- Semi-supervised learning
- Failures
- Generative adversarial networks (learning the loss)

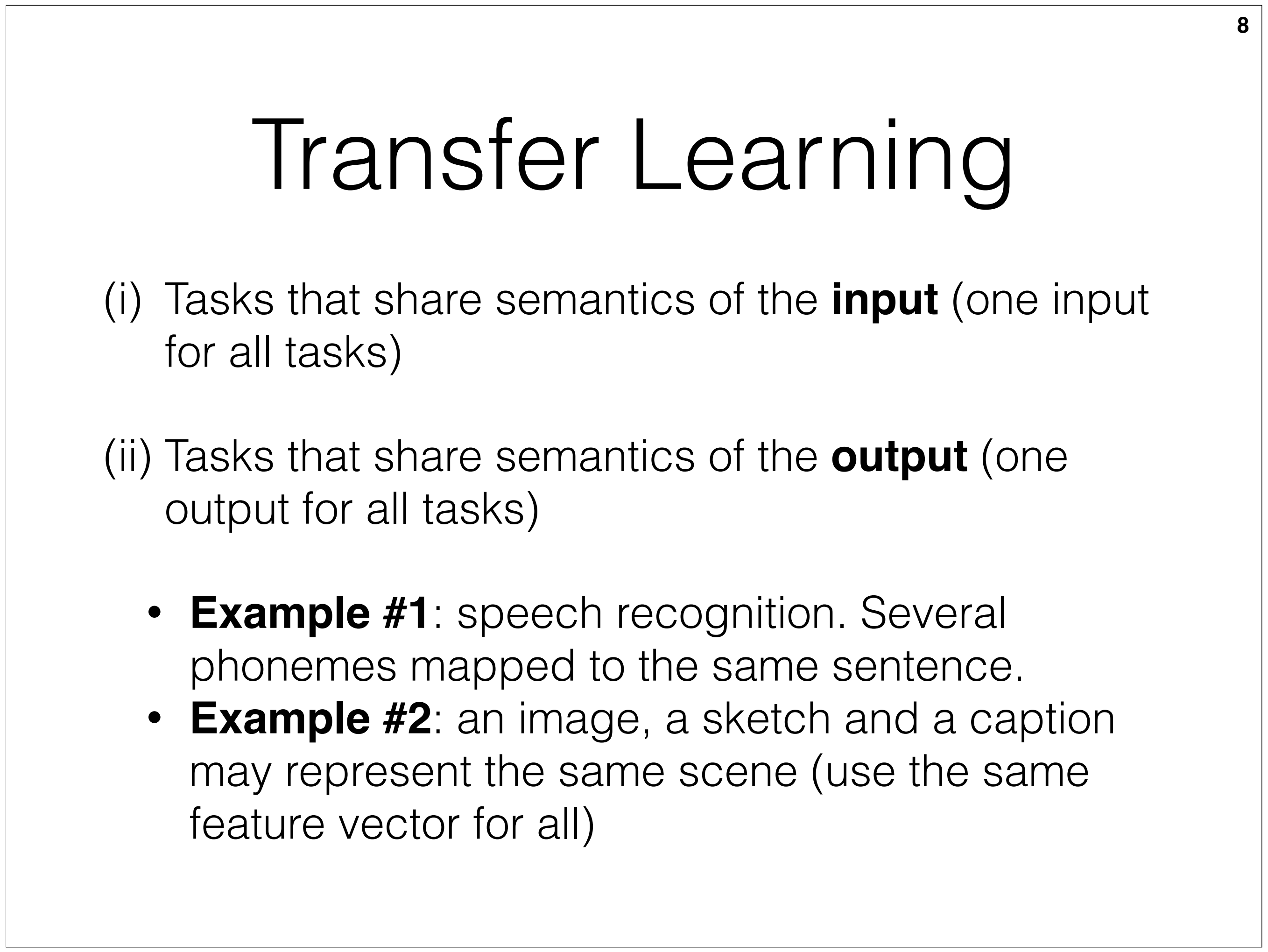
Notes



Transfer Learning

- The learner performs multiple tasks
- Learn a representation for $p_1(x)$ and use it to learn a representation for $p_2(z)$
- We assume that many factors of variations in the data x and z are **shared**
- **Example**: x are images of cats and dogs, and z are images of ants and wasps
  They share low-level notions, such as edges and visual shapes, geometrical and illumination effects

In transfer learning we consider two data domains that have common factors of variations and use concepts that were learned in one domain to help to learn concepts in the other domain.



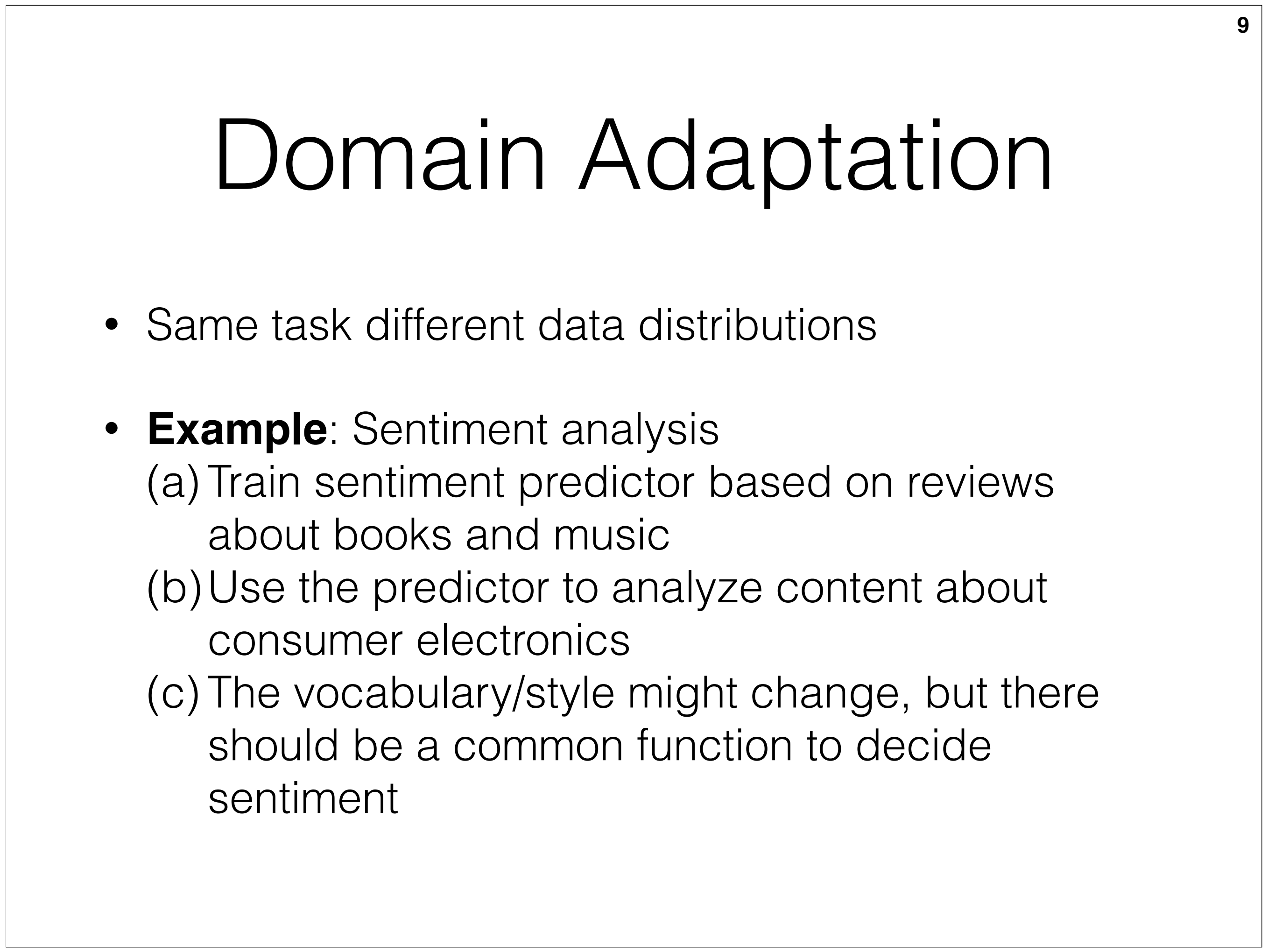
Transfer Learning

(i) Tasks that share semantics of the **input** (one input for all tasks)

(ii) Tasks that share semantics of the **output** (one output for all tasks)

- **Example #1**: speech recognition. Several phonemes mapped to the same sentence.
- **Example #2**: an image, a sketch and a caption may represent the same scene (use the same feature vector for all)

Transfer learning can be applied to tasks that share semantics either at the input or at the output. We have already seen the case of tasks that share the semantics of the input (e.g. use one image to perform many tasks, such as classification, detection, segmentation, 3D reconstruction etc).
Let us focus more on the second case.
In the case of tasks that share the semantics of the output, it is useful to share upper layers (near the output)



Domain Adaptation

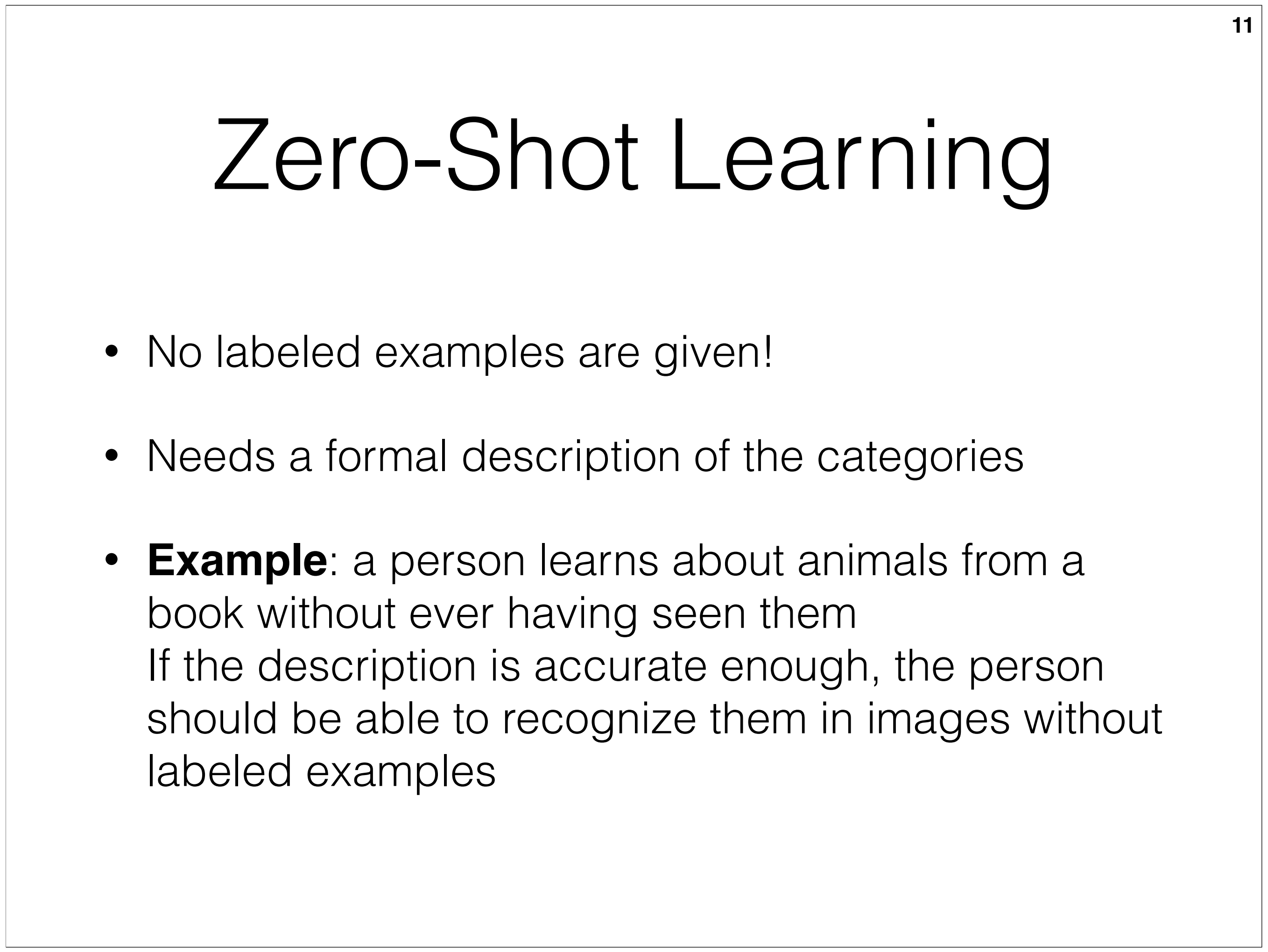
- Same task different data distributions
- **Example**: Sentiment analysis
  - (a) Train sentiment predictor based on reviews about books and music
  - (b) Use the predictor to analyze content about consumer electronics
  - (c) The vocabulary/style might change, but there should be a common function to decide sentiment

Sentiment analysis consists in determining whether a comment expresses a positive or a negative sentiment.

One-Shot Learning

- Only one labeled example is given!
- Humans seem to learn with very few labeled examples
- What we have learned on other categories allows us to separate data into factors of variation
- Those factors are useful to classify new categories

An extreme form of transfer learning



Zero-Shot Learning

- No labeled examples are given!
- Needs a formal description of the categories
- **Example**: a person learns about animals from a book without ever having seen them
  If the description is accurate enough, the person should be able to recognize them in images without labeled examples

An even more extreme form of transfer learning

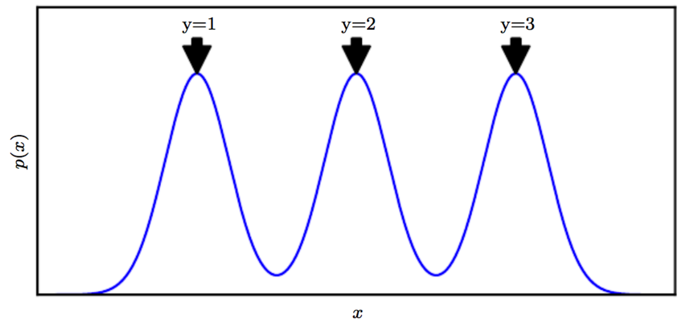
Semi-Supervised Disentangling of Causal Factors

- What makes one representation better than another?
  - Features correspond to separate causes
  - Features are easy to model (e.g., sparse, independent)

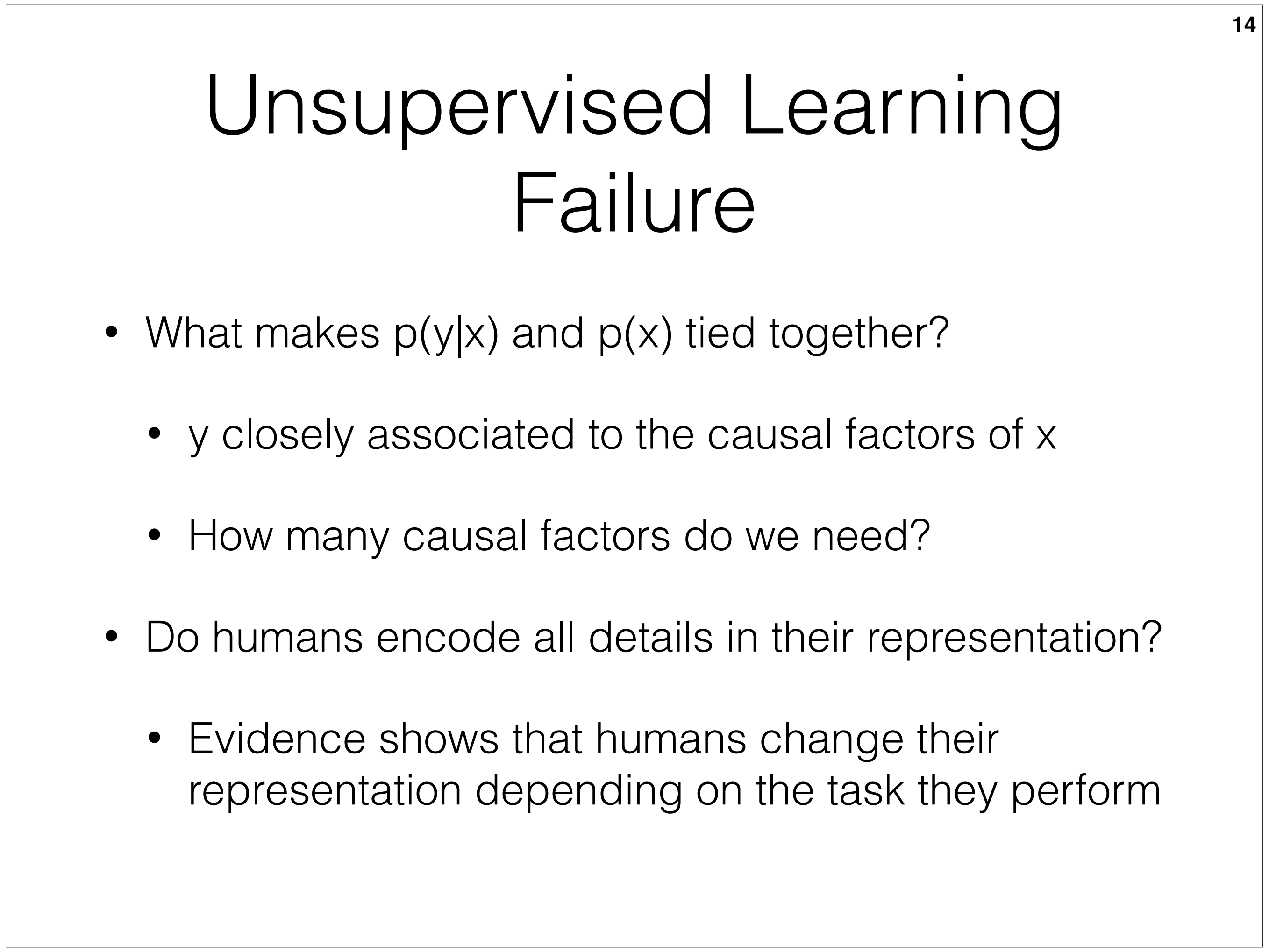
Note

Unsupervised Learning

- **Failure example**: p(x) is uniform, find f(x) = E[y|x] — A training set of x does not give us information about p(y|x)
- **Success example**: p(x) a mixture with well-separated Gaussians (one for each y instance) — then p(x) will capture p(y|x) very well



Suppose that we first learn p(x) and then we look for a representation for p(y|x).
When does learning p(x) not help to learn p(y|x)?
This might not work if y is not among the causes of x.
Use marginalization to obtain p(x) from the conditional distribution, then p(x) = \sum_y p(x,y)
We want p(y|x) = p(x,y)/p(x)
If p(x) has separate components for each x, this means that p(x,y) has also separate components that do not overlap with marginalization. A single sample is sufficient to learn p(y|x).
In this case disentangling becomes feasible.



Unsupervised Learning Failure

- What makes p(y|x) and p(x) tied together?
  - y closely associated to the causal factors of x
  - How many causal factors do we need?
- Do humans encode all details in their representation?
  - Evidence shows that humans change their representation depending on the task they perform

When y is associated to the causal factors of x then UL helps semi-supervised learning. UL needs to disentangle the factors of variation.

Unsupervised Learning Failure

- **Example**: Let h be all the causal factors of x and let y be one of them
- p(h,x) = p(x|h) p(h)
- p(x) = E_h[p(x|h)]
- Suppose y = h_i, but we do not know which one
- Solution: Predict all h_j and predict y from h

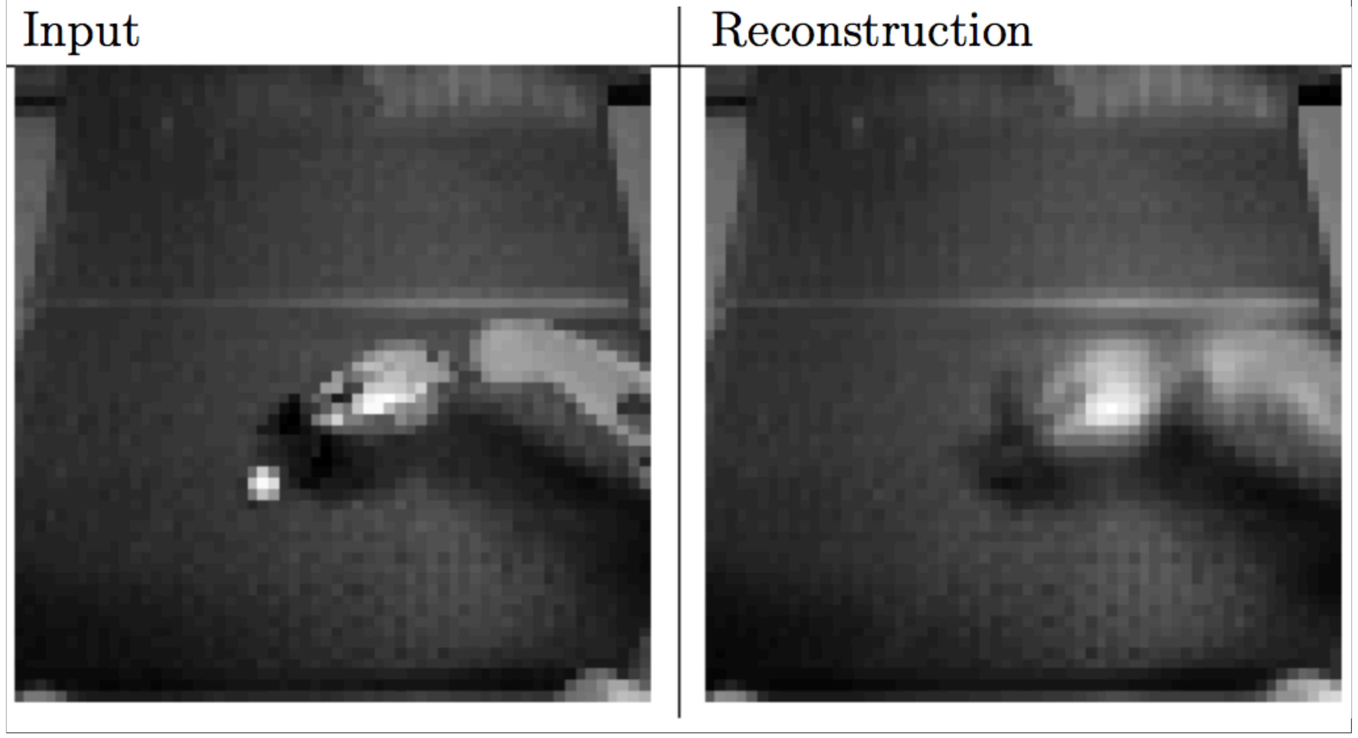
Disentanglement of factors in h makes it easy to predict the target y = h_i.

Unsupervised Learning Failure

- Another way to change the representation is to change the loss function (eg, L^2 vs entropy)
- For example, L^2 says that factors of variations are important only when they lead to a change of brightness
- This is not a good choice if we are interested in small objects

Note

Unsupervised Learning Failure



Autoencoder trained with L2 loss. The reconstruction fails to pick up the ping pong ball.

Learning the Loss

- We might not know the best loss function for a task beforehand
- One solution is then to **learn** it
- **Generative adversarial networks** (GAN) provide one such framework

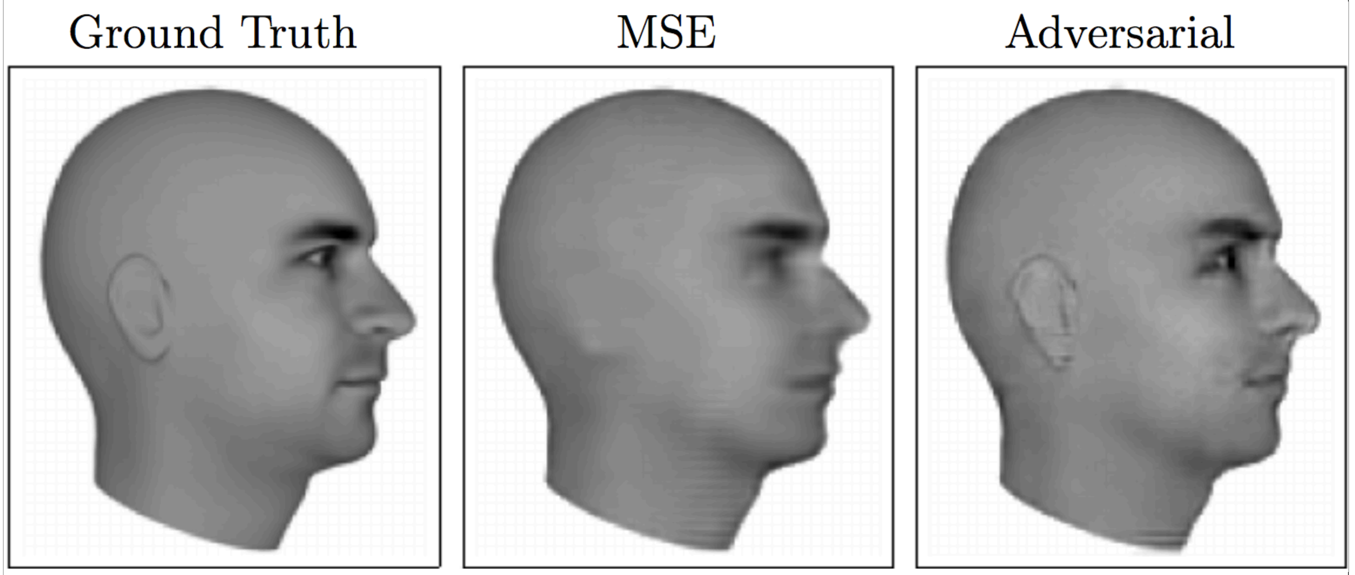
Note

Generative Adversarial Networks

- We are given a training dataset $x_1,\dots,\ x_m$
- Two models: a **generator G** and a **discriminator D**
- **D** tries to distinguish samples from the training set from samples from **G** (it is a binary classifier)
- **G** generates samples from random Gaussian noise and tries to fool **D**
- Equilibrium is reached when G wins

Note

Generative Adversarial Networks



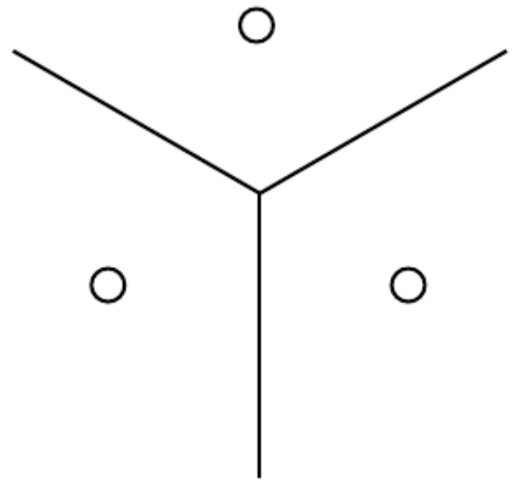
The MSE loss does not emphasize the ear because it does not have strong gradients. GAN instead learns to pick up the ear as a salient feature (omitting ears could be used to discriminate real samples from generated/fake ones).

Models

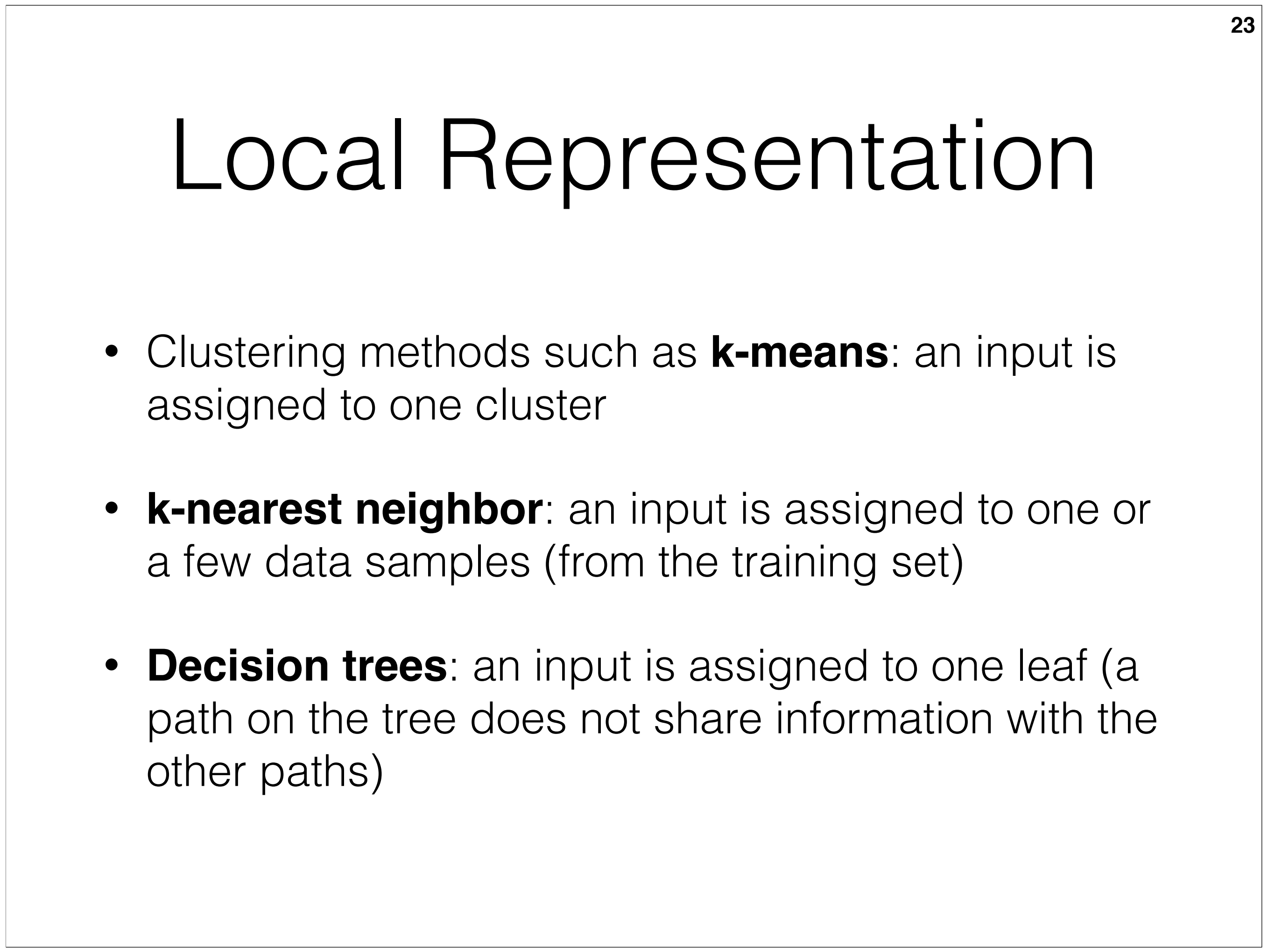
- Local vs distributed representations

Note

Local Representation



Nearest neighbour is an example of a non distributed representation. With N feature detectors there are N regions in space. This representation is a one-hot encoding (1 case per feature). The advantage is a simple optimization (one output for each region). The disadvantage is that generalization is limited. Only based on local smoothness.



Local Representation

- Clustering methods such as **k-means**: an input is assigned to one cluster
- **k-nearest neighbor**: an input is assigned to one or a few data samples (from the training set)
- **Decision trees**: an input is assigned to one leaf (a path on the tree does not share information with the other paths)

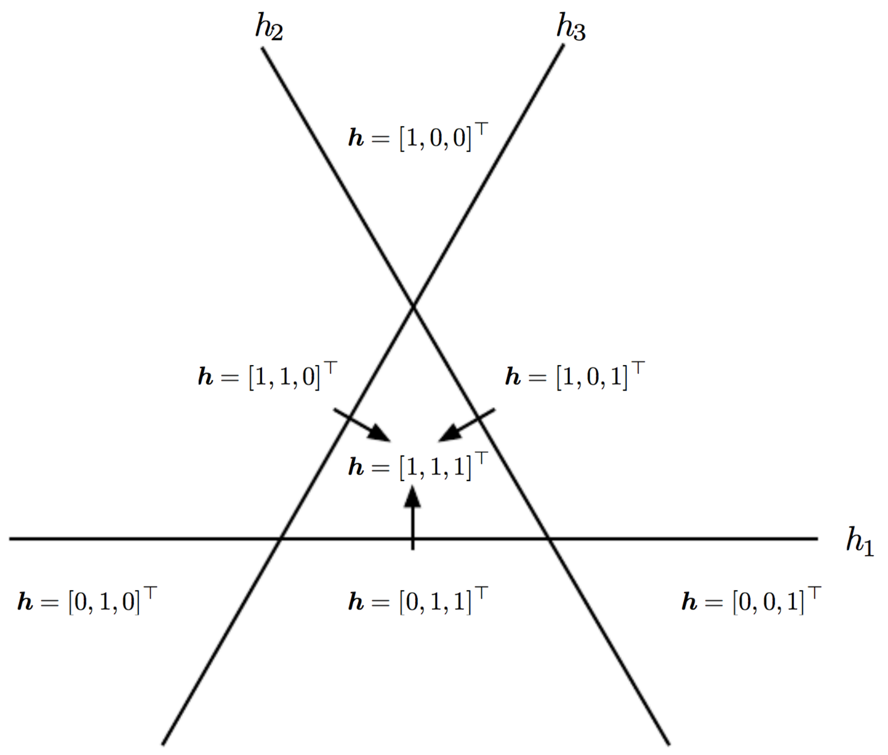
Note

Local Representation

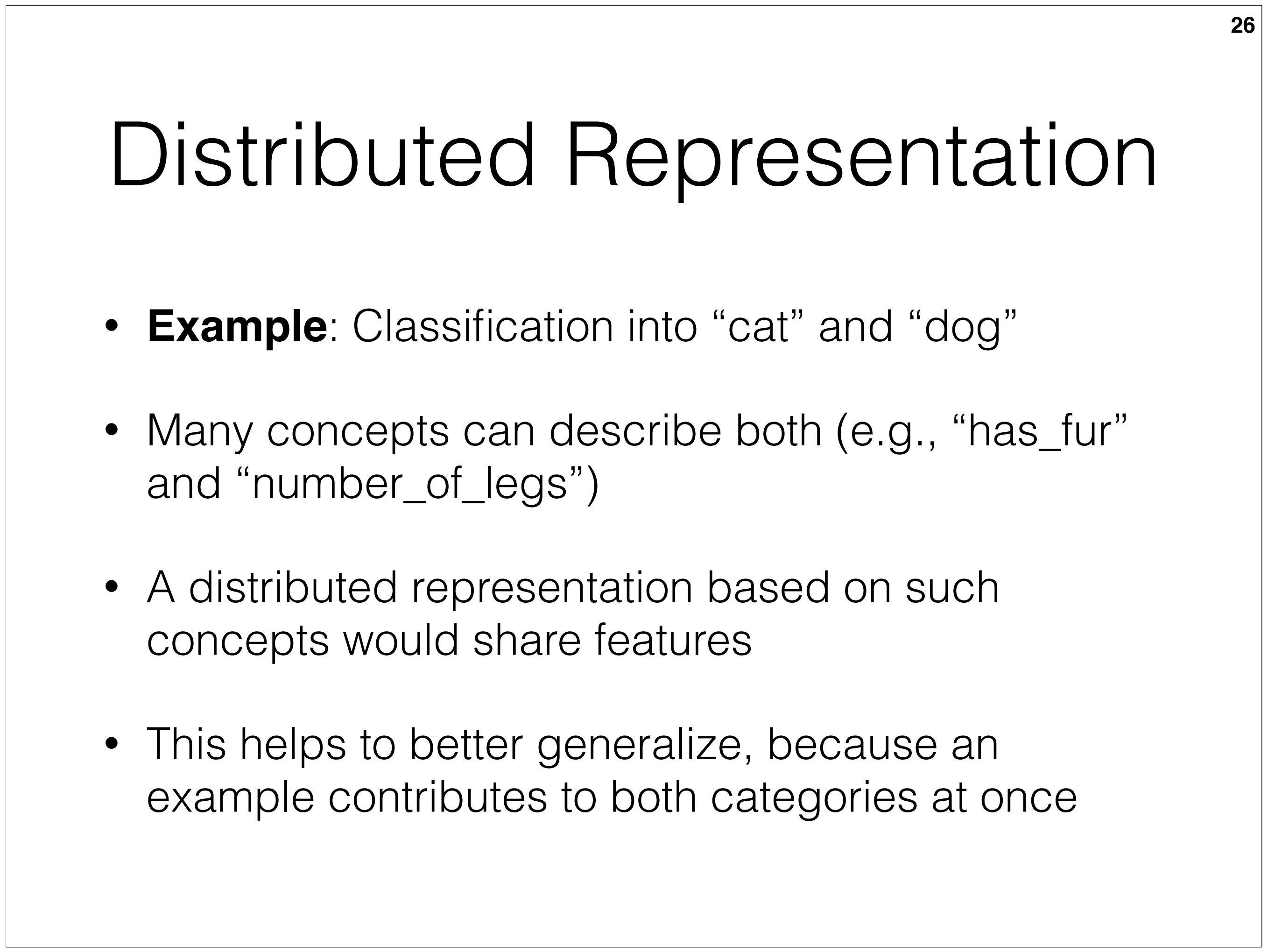
- Relies on **smoothness constraint**: small input changes should give small output changes
- Because each input/output mapping is independent from the others, this representation **does not generalize** to new inputs

Note

Distributed Representation


h_2
h_3
h_1
$\boldsymbol{h} = [1,0,0]^\top$
$\boldsymbol{h} = [1,1,0]^\top$
$\boldsymbol{h} = [1,0,1]^\top$
$\boldsymbol{h} = [1,1,1]^\top$
$\boldsymbol{h} = [0,1,0]^\top$
$\boldsymbol{h} = [0,1,1]^\top$
$\boldsymbol{h} = [0,0,1]^\top$

In a distributed representation regions capture a different configuration (a pattern of activity). N features can describe an exponential number of regions. The representation tries to capture different causal factors as separate features so that a classifier built on top of these features can just focus on a subset of them.



Distributed Representation

- **Example**: Classification into “cat” and “dog”
- Many concepts can describe both (e.g., “has_fur” and “number_of_legs”)
- A distributed representation based on such concepts would share features
- This helps to better generalize, because an example contributes to both categories at once

Distributed representations thus induce a rich similarity space. The sharing of concepts tends to make the similarity more semantically driven.

Distributed Representation

- Consists of a set of elements that can be set separately from each other
- A data sample is then represented by a pattern of activity distributed over each element

Note

Distributed Representation

- Introduces a gain when representing complex structures with a small number of parameters
- Consider n d-dimensional features, then they can distinguish $O(n^d)$ regions through thresholding with $O(nd)$ parameters
- A local representation would require $O(n^d)$ parameters
- Generalization favors the distributed representation

Note

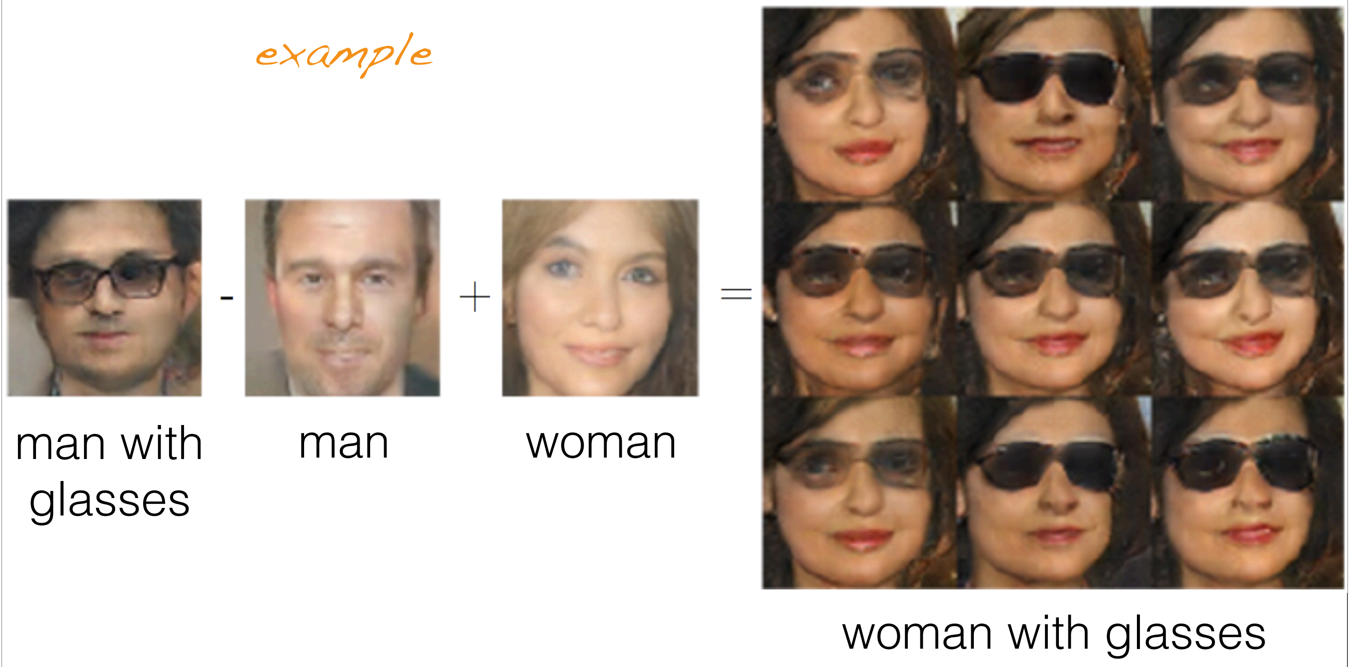
Distributed Representation

- The capacity is limited although the encodings are exponentially many
- Based on the assumption that data is not structured in any possible random way, but favors shared substructures
- This seems to be the case for images for example

Note

Distributed Representation

example



In the work of Radford et al (2015) —DCGAN— they learned a representation without labels, that naturally identifies the concepts of gender and of wearing sunglasses.