– Tutorial: High Dimensional Computing –

Vector Encodings of Real World Data

Stefan Schubert

TECHNISCHE UNIVERSITÄT CHEMNITZ

Example:

- Database of stations along a rail route
    - Name
    - Arrival times
    - Image

Source: NRKbeta.no, http://nrkbeta.no/2013/01/15/nordlandsbanen-minute-by-minute-season-by-season/

Encoding in a Hypervector H:

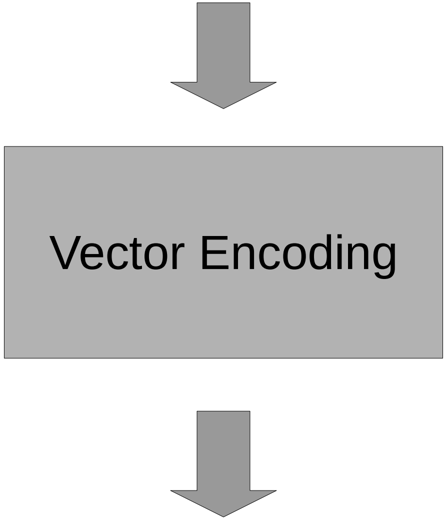
$$H = \text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3$$

To do calculations with this formula …

… we need actual hypervectors!

Where come the HD vectors from?

$$H = \mathrm{name} \otimes \mathrm{station1} + 12 \otimes \mathrm{arrival} + \mathrm{image} \otimes I_3$$



$$H_{\mathrm{name}} \quad H_{\mathrm{station1}} \quad H_{12} \quad H_{\mathrm{arrival}} \quad H_{\mathrm{image}} \quad H_{I_3}$$

Outline

1) Requirements for vector encoding
2) Encodings for real world data
    1) Encodings for finite sets
    2) Integer Encodings
    3) Image encodings

Requirements for vector encoding

1) Distributed representations …
- for VSAs that depend on element-wise operations
- for robustness against distortions / bit-flips

Example: Encoding of integers i from 0 to 10

$$i = 3$$

Disturb 1st bit →

Base-2 numeral system:	$H_3 = (00000000011)$	$H_3 = (10000000011)$	1027
One-hot:	$H_3 = (00010000000)$	$H_3 = (10010000000)$	undefined
Distributed:	$H_3 = (10110011100)$	$H_3 = (00110011100)$	Dependent on encoding & #dimensions

Requirements for vector encoding

1) Distributed representations …
   - for VSAs that depend on element-wise operations
   - for robustness against distortions / bit-flips
2) HD vectors
   - robust to noise
   - capacity for different amount of information (from number to whole program)

Requirements for vector encoding

1) Distributed representations …
   - for VSAs that depend on element-wise operations
   - for robustness against distortions / bit-flips
2) HD vectors
   - robust to noise
   - capacity for different amount of information (from number to whole program)

3a) (Almost) orthogonal vectors
   - for retrievals

$$\text{arrival?} \leftarrow H \otimes 12$$

$$\text{arrival?} \leftarrow (\text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3) \otimes 12$$

$$\text{arrival?} \leftarrow \underbrace{\text{name} \otimes 12 \otimes \text{station1}}_{\text{noise}} + \underbrace{12 \otimes 12}_{\text{identity}} \otimes \text{arrival} + \underbrace{\text{image} \otimes 12 \otimes I_3}_{\text{noise}}$$

$$\text{arrival?} \leftarrow \text{arrival} + noise$$

Requirements for vector encoding

1) Distributed representations …
   - for VSAs that depend on element-wise operations
   - for robustness against distortions / bit-flips
2) HD vectors
   - robust to noise
   - capacity for different amount of information (from number to whole program)

3a) (Almost) orthogonal vectors
   - for retrievals

3b) HD vectors that encode meaningful data
   - e.g., integers
   - potentially preserve similarity

Requirements for vector encoding

What do we need?

…

3b) HD vectors that encode meaningful data

- e.g., integers
- potentially **preserve similarity**

Integer i:	0	1	2	3
	$\begin{pmatrix}1\\0\\0\\1\\0\\1\end{pmatrix}$	$\begin{pmatrix}0\\0\\1\\1\\0\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\1\\0\\0\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\\1\\0\\1\\0\end{pmatrix}$
sim(0, i):	1	0.666…	0.333…	0

Requirements for vector encoding

What do we need?

…

3b) HD vectors that encode meaningful data

- e.g., integers
- potentially **preserve similarity**

$$\text{arrival?} \leftarrow H \otimes 13$$
$$\text{arrival?} \leftarrow (\text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3) \otimes 13$$
$$\text{arrival?} \leftarrow \text{name} \otimes 13 \otimes \text{station1} + 12 \otimes 13 \otimes \text{arrival} + \text{image} \otimes 13 \otimes I_3$$
$$\text{arrival?} \leftarrow noise + \underbrace{12 \otimes 13}_{} \otimes \text{arrival} + noise$$

Trade-off!

If we preserve similarity between 12 & 13, we retrieve a vector similar to “arrival”

Arrival? ← ~arrival / yes

If 12 & 13 are almost orthogonal, we retrieve a *noise* vector

Arrival? ← noise / no

Requirements for vector encoding

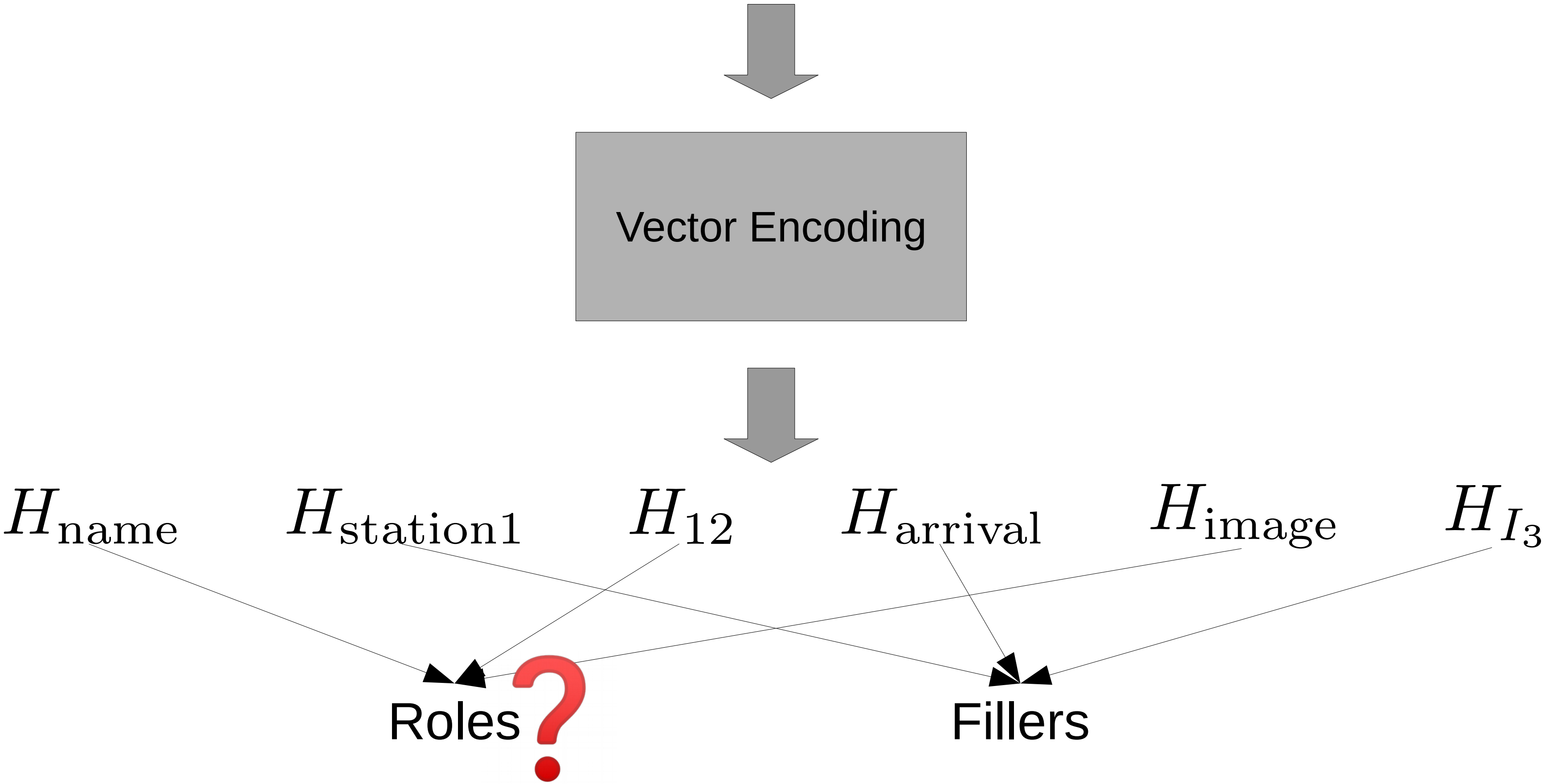
1) Distributed representations …
   - for VSAs that depend on element-wise operations
   - for robustness against distortions / bit-flips
2) HD vectors
   - robust to noise
   - capacity for different amount of information (from number to whole program)

3a) (Almost) orthogonal vectors
   - for retrievals

3b) HD vectors that encode meaningful data
   - e.g., integers
   - potentially preserve similarity

Trade-off: Similarity preserving encodings influence retrievals!

$$H = \text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3$$



Encoding for roles

- Finite set of roles
- Encoding: Draw random HD vectors; store in item memory
- They are (likely almost) orthogonal
- Guarantees retrieval with roles in HD vectors:

$$name? \leftarrow H \otimes \text{name}$$
$$name? \leftarrow (\text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3) \otimes \text{name}$$
$$name? \leftarrow \underbrace{\text{name} \otimes \text{name}}_{\text{identity}} \otimes \text{station1} + \underbrace{12 \otimes \text{name} \otimes \text{arrival}}_{\text{noise}} + \underbrace{\text{image} \otimes \text{name} \otimes I_3}_{\text{noise}}$$
$$name? \leftarrow \text{station1} + noise$$

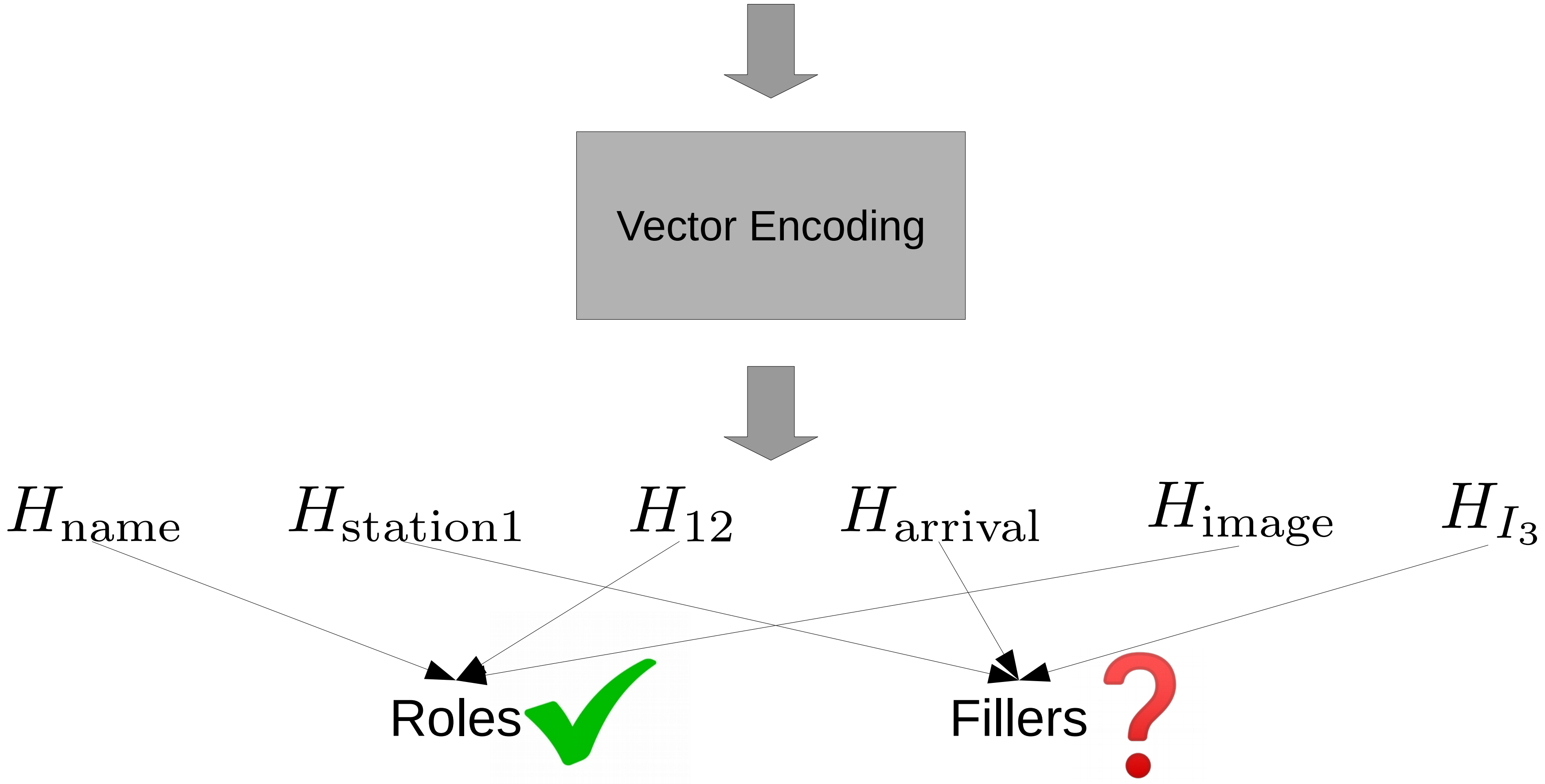
Kanerva (1997). Fully Distributed Representation. Proc. of Real World Computing Symp.

Levy et al. (2013). Learning Behavior Hierarchies via High-Dimensional Sensor Projection. Proc. of AAAI Conf. on Learning Rich Representations from Low-Level Sensors

Kleyko et al. (2015). Imitation of honey bees' concept learning processes using Vector Symbolic Architectures. Biologically Inspired Cognitive Architectures

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

$$H = \text{name} \otimes \text{station1} + 12 \otimes \text{arrival} + \text{image} \otimes I_3$$



How to encode fillers / real world data?

There aren‘t general approaches, that work for each problem!

- Depends
  - on the chosen VSA / HD vector
  - on type of real world data, e.g.
    - Finite set, e.g. names, alphabet, categories, symbols
    - Special type, e.g. weekday
    - Number, e.g. age, range, …
    - Image, e.g. for place recognition, for image classification, …
    - …
  - on similarity preservation

Encodings for arbitrary data into arbitrary HD vectors are an open question

Encodings for

1) Finite Sets   2) Integers   3) Images (for place rec.)

For a set with M elements:

1) Draw M random vectors

Kanerva (1997). Fully Distributed Representation. Proc. of Real World Computing Symp.

Levy et al. (2013). Learning Behavior Hierarchies via High-Dimensional Sensor Projection. Proc. of AAAI Conf. on Learning Rich Representations from Low-Level Sensors

Kleyko et al. (2015). Imitation of honey bees' concept learning processes using Vector Symbolic Architectures. Biologically Inspired Cognitive Architectures

2) Draw one random vector and apply circular shift (M-1)-times

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

2) Draw one random vector and apply circular shift (M-1)-times:

$$H_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix} \quad H_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \quad H_3 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ \vdots \end{pmatrix}$$

Works due to local operations of VSAs

Encodings for

1) Finite Sets 2) Integers 3) Images (for place rec.)

For a set with M elements:

1) Draw M random vectors

Kanerva (1997). Fully Distributed Representation. Proc. of Real World Computing Symp.

Levy et al. (2013). Learning Behavior Hierarchies via High-Dimensional Sensor Projection. Proc. of AAAI Conf. on Learning Rich Representations from Low-Level Sensors

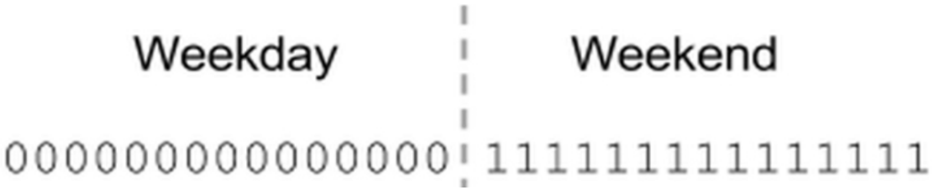
Kleyko et al. (2015). Imitation of honey bees' concept learning processes using Vector Symbolic Architectures. Biologically Inspired Cognitive Architectures

2) Draw one random vector and apply circular shift (M-1)-times

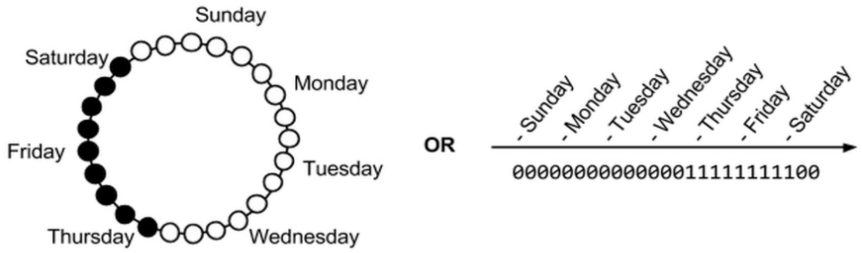
Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Sparse binary

1) Categorical data:



2) Circular categorical data:



Purdy (2016). Encoding Data for HTM Systems. CoRR abs/1602.05925 (2016)

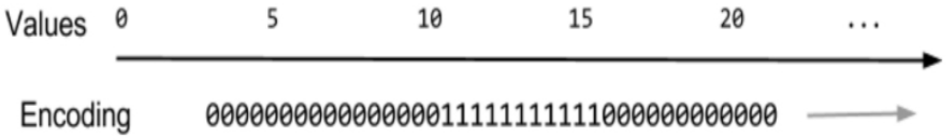
Encodings for

1) Finite Sets 2) Integers 3) Images (for place rec.)

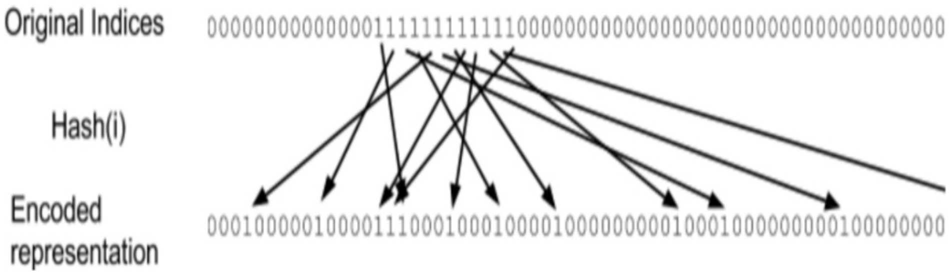
Sparse binary

Dense binary

1) Simple:



2) More flexible:



Purdy (2016). Encoding Data for HTM Systems.
CoRR abs/1602.05925 (2016)

Encodings for

1) Finite Sets 2) Integers 3) Images (for place rec.)

Sparse binary

For M integers:

Orthogonal

1) Draw M random vectors
2) Draw 1 random vector and perform
   M-1 circular shifts

Distance Preserving

3) "Linear Mapping"
4) Approximate "Linear Mapping"

Dense binary

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Encodings for
1) Finite Sets 2) Integers 3) Images (for place rec.)

Sparse binary

For M integers:

Orthogonal

1) Draw M random vectors
2) Draw 1 random vector and perform M-1 circular shifts

Distance Preserving

3) “Linear Mapping”
4) Approximate “Linear Mapping”

Dense binary

2) Draw one random vector and apply circular shift (M-1)-times:

$$H_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix} \quad H_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix} \quad H_3 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ \vdots \end{pmatrix}$$

Works due to local operations of VSAs

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Encodings for

1) Finite Sets 2) Integers 3) Images (for place rec.)

Sparse binary

For M integers:

Orthogonal

1) Draw M random vectors
2) Draw 1 random vector and perform
   M-1 circular shifts

Distance Preserving

3) "Linear Mapping"
4) Approximate "Linear Mapping"

Dense binary

More similar integers are more similar in HD space

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Encodings for
1) Finite Sets 2) Integers 3) Images (for place rec.)

Sparse binary

For M integers:

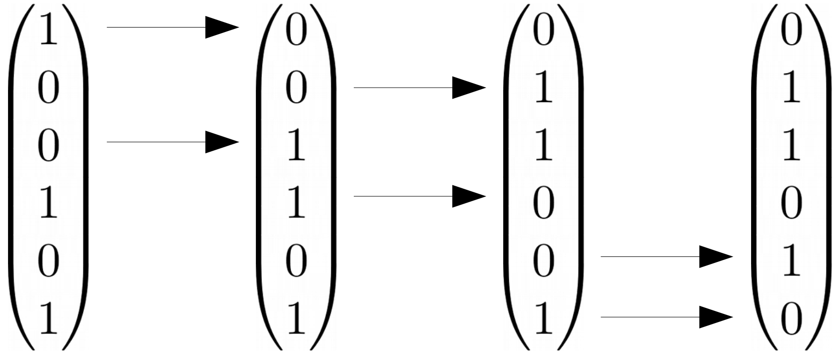
Orthogonal
1) Draw M random vectors
2) Draw 1 random vector and perform M-1 circular shifts

Distance Preserving
3) "Linear Mapping"
4) Approximate "Linear Mapping"

Dense binary

- Exact:
  1) Draw 1 random HD vector
  2) For each new integer:
     1) Flip 0/1-bits from the first HD vector that haven't been processed, yet

i: 0 1 2 3



sim(0, i): 1 0.666… 0.333… 0

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Encodings for
1) Finite Sets 2) Integers 3) Images (for place rec.)

Sparse binary

For M integers:

Orthogonal

1) Draw M random vectors
2) Draw 1 random vector and perform M-1 circular shifts

Distance Preserving

3) "Linear Mapping"
4) Approximate "Linear Mapping"

Dense binary

- Approximate:
  1) Draw 2 random HD vectors for first and last integer
  2) For integers in between: Concatenate parts from first & last vector

i: 0 1 2 3

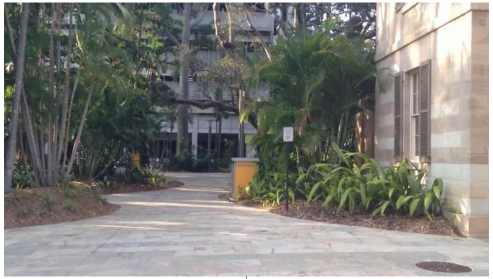
$$\begin{pmatrix}1\\0\\0\\1\\0\\1\end{pmatrix} \quad \begin{pmatrix}1\\0\\0\\1\\1\\0\end{pmatrix} \quad \begin{pmatrix}1\\0\\1\\0\\1\\0\end{pmatrix} \quad \begin{pmatrix}0\\1\\1\\0\\1\\0\end{pmatrix}$$

sim(0, i): 1 0.666… 0.333… 0

Kleyko et al. (2018). Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. Trans. on Neural Networks and Learning Systems

Encodings for

1) Finite Sets 2) Integers **3) Images (for place rec.)**



A. Glover (2014). Day and night with lateral pose change datasets

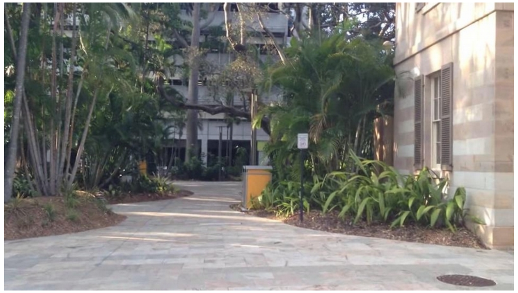
CNN

$$D \in \mathbb{R}^N$$

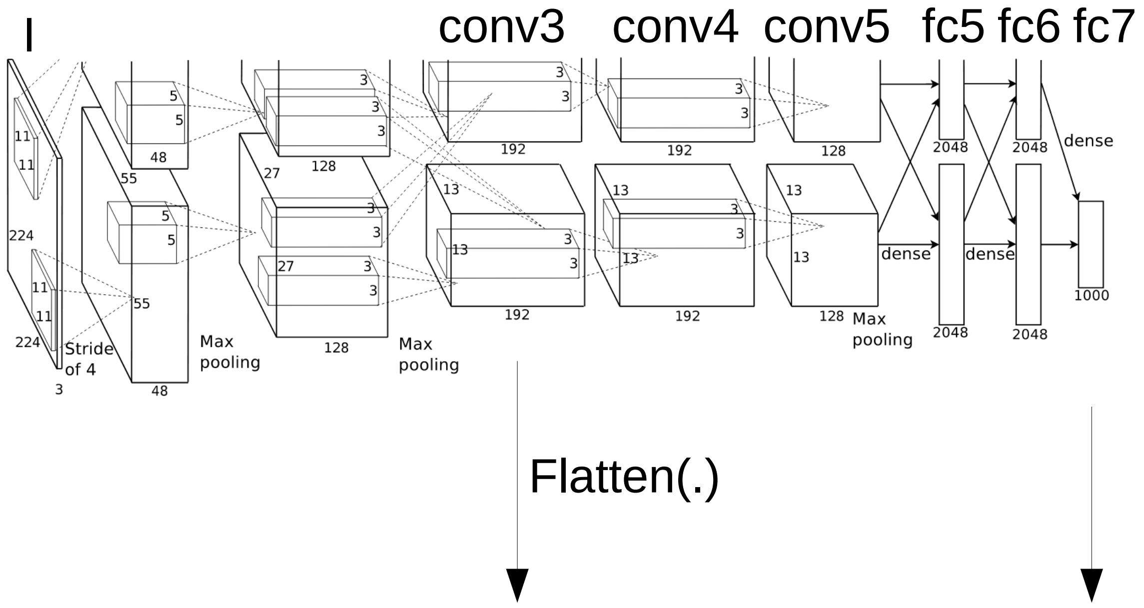
Requirement from place recognition:

- **For a given image, we need an HD vector encoding that preserves similarity to images of same places**
- **CNNs can be used to generate suited HD vectors for place recognition**

Using AlexNet for Place Recognition



A. Glover (2014). Day and night with lateral pose change datasets



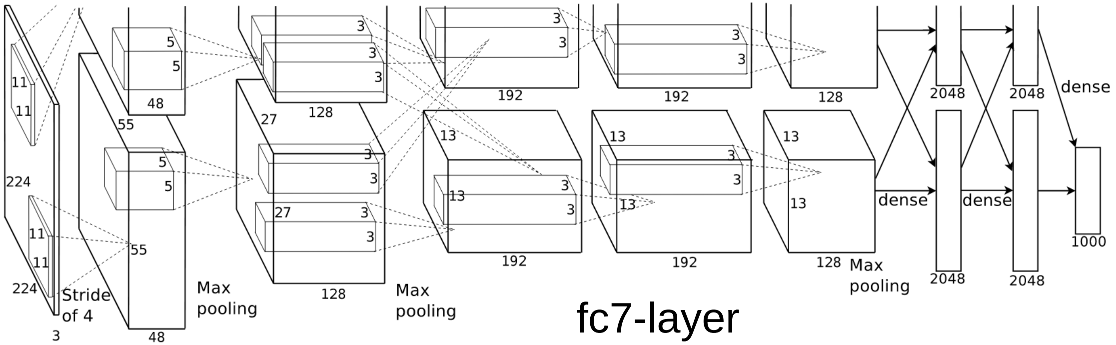
$\mathbb{R}^{64896}$ $\mathbb{R}^{1000}$

A. Krizhevsky, I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. in Advances in Neural Information Processing Systems

Sünderhauf et al. (2015). On the performance of ConvNet features for place recognition. Int. Conf. on Intelligent Robots and Systems

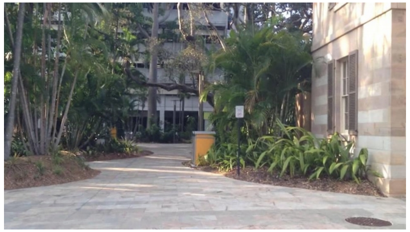
Using AlexNet for Place Recognition

I conv3 conv4 conv5 fc5 fc6 fc7



fc7-layer
performs better for
viewpoint changes

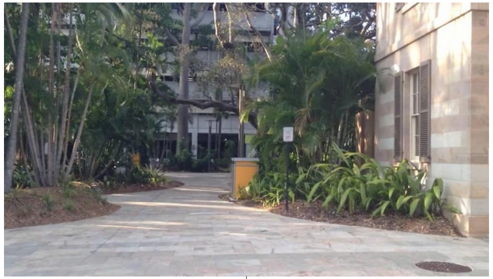
conv3-layer
performs better for
appearance changes



A. Glover (2014). Day and night with lateral pose change datasets

A. Krizhevsky, I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. in Advances in Neural Information Processing Systems

Sünderhauf et al. (2015). On the performance of ConvNet features for place recognition. Int. Conf. on Intelligent Robots and Systems



A. Glover (2014). Day and night with lateral pose change datasets

CNN

$D \in \mathbb{R}^N$

Raw

$H := D$

$H \in \mathbb{R}^N$

Normalization

$$H := \frac{D}{\|D\|_2}$$

$H \in [-1, 1]^N$

LSH
(Locality Sensitve Hashing)

$$H := \frac{D^T}{\|D\|_2} \cdot P$$

$H \in [-1, 1]^n, n < N$

sLSBH
(sparse Locality Sensitive Binary Hashing)

1) LSH
2) Find the d-th highest value max_d and the d-th lowest value min_d in D^T*P
3) Concat

$$H := \begin{bmatrix} (D^T \cdot P) \geq max_d \\ (D^T \cdot P) \leq min_d \end{bmatrix}$$

$H \in \{0, 1\}^{2n}$

$d := \|H\|_1 / 2$

Neubert, Schubert, Protzel (2019). A Neurologically Inspired Sequence Processing Model for Mobile Robot Place Recognition. Robotics and Automation Letters (RA-L)