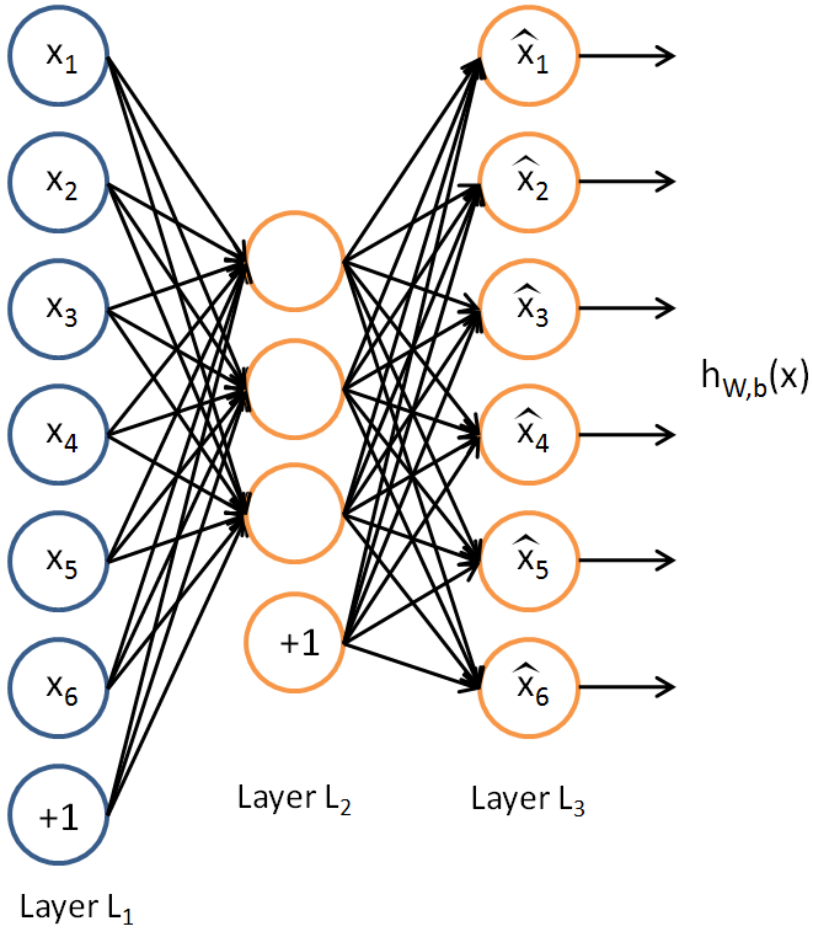
More Neural Networks

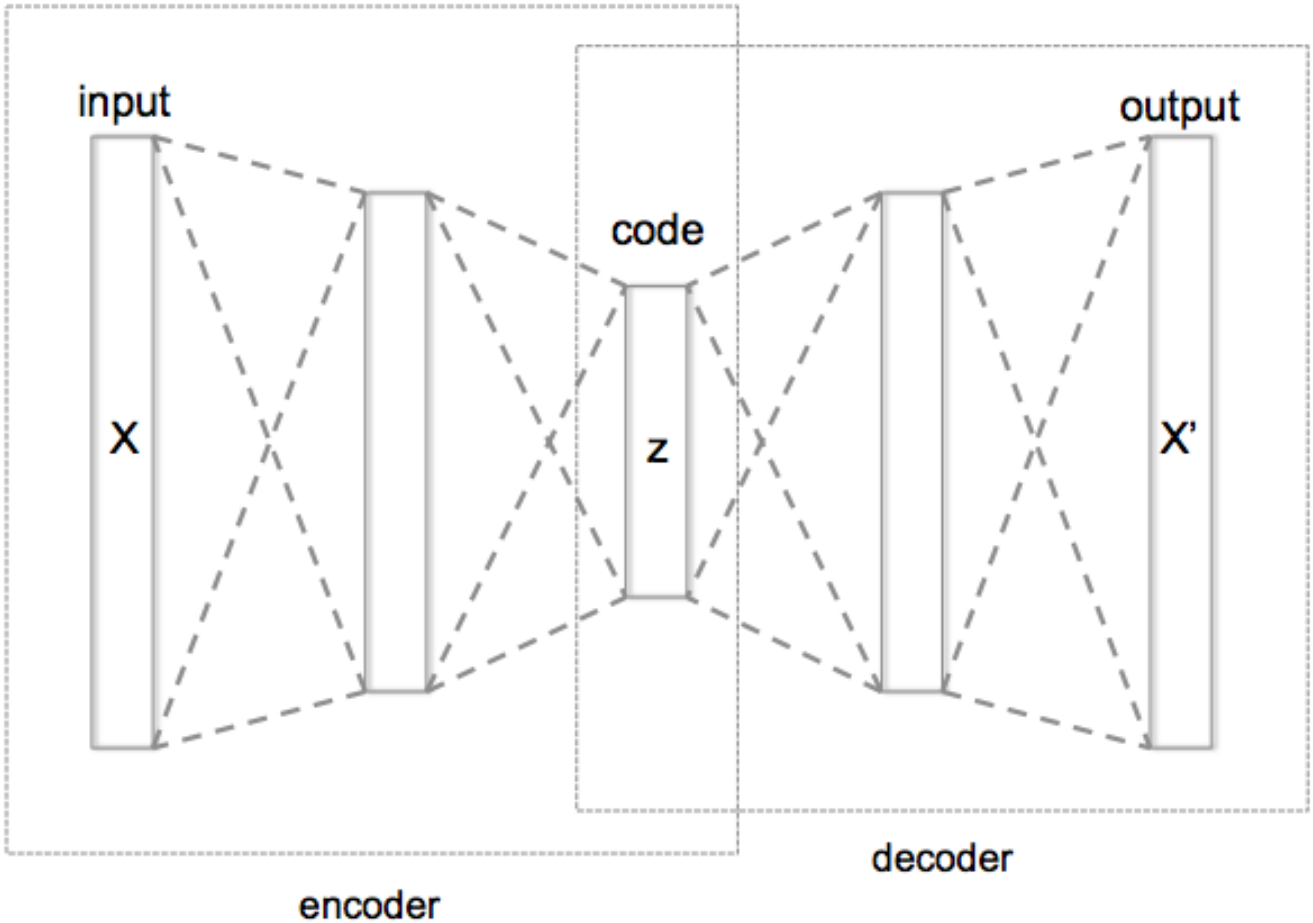
Mladen Kolar (mkolar@chicagobooth.edu)

Autoencoder



Network trained to reproduce its input at the output layer.
Usually tie the weights that go into and out of the hidden layer.

Autoencoder



https://bradleyboehmke.github.io/HOML/autoencoders.html

Autoencoder

Loss function

- For real valued inputs, try to find weights such that

$$\frac{1}{2}\sum_{k}(x_k - \hat{x}_k)^2$$

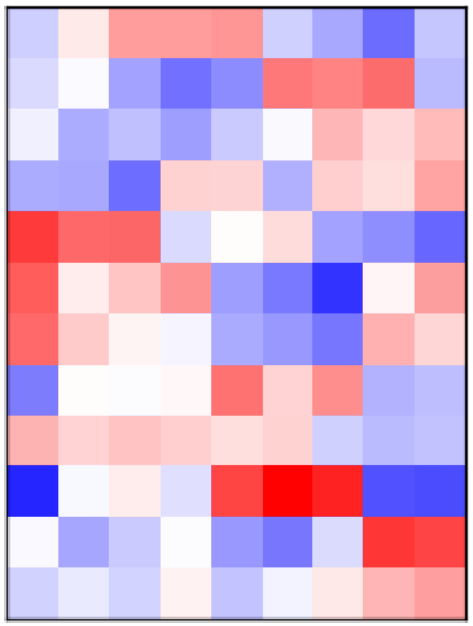
  is minimized
- For binary input cross entropy is used, which is similar to deviance

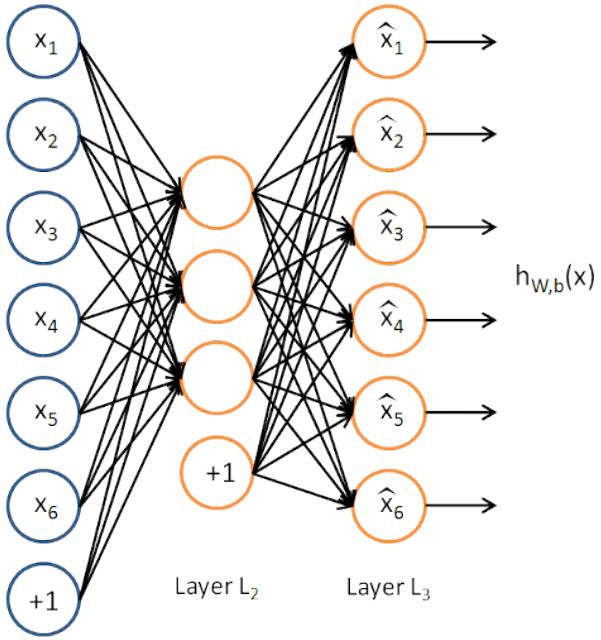
Fitting autoencoder

- Same tricks as before
- Greedy learning of stacked autoencoders
- https://www.cs.toronto.edu/~hinton/science.pdf

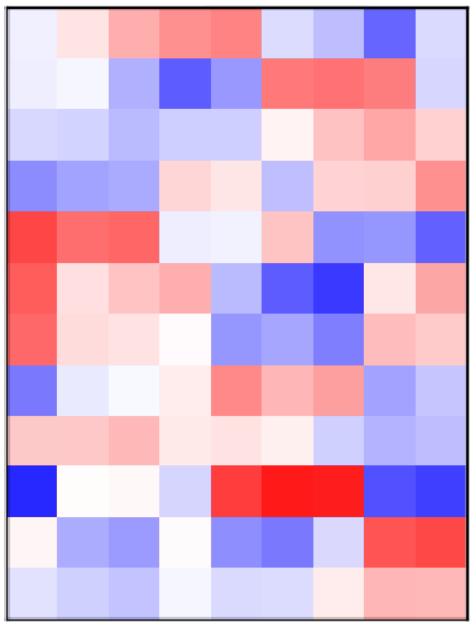
Autoencoder

Original Data



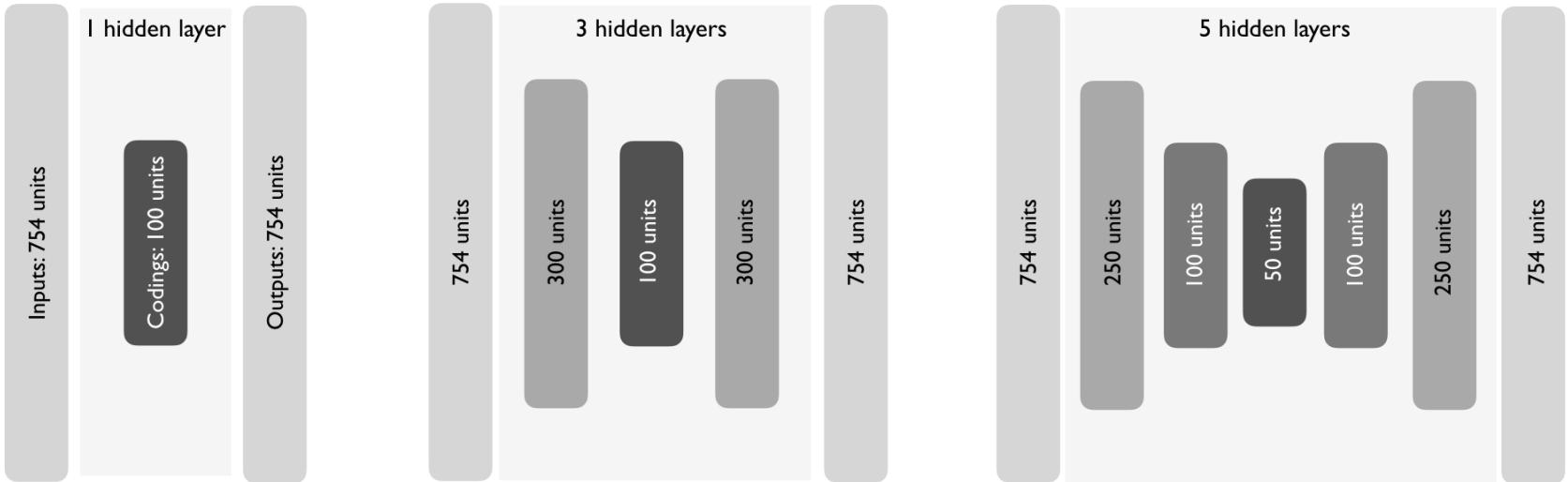


Reconstruction



$$\min \sum_{i=1}^{n} \|x_i - \sigma(W'^{\top}\sigma(Wx_i + b) + b')\|_2^2$$

Stacked (or deep) autoencoders

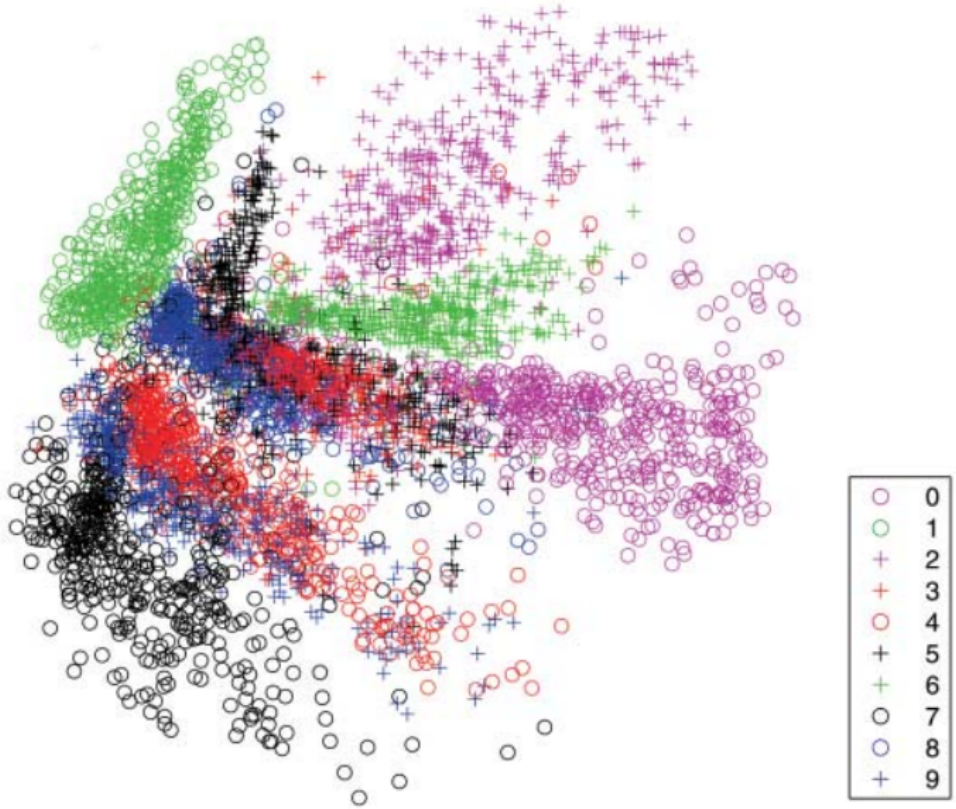


Additional depth can help

- codes represent more complex, nonlinear relationships
- reduced computational cost
- better data compression than shallower, or linear autoencoders.

Autoencoder: Why are they useful?

Learning compressed representation of the input distribution (dimensionality reduction)



Autoencoder structure: 784 — 1000 — 500 — 250 — 2

https://www.cs.toronto.edu/~hinton/science.pdf

A comparison of methods for compressing digit images to 30 real numbers

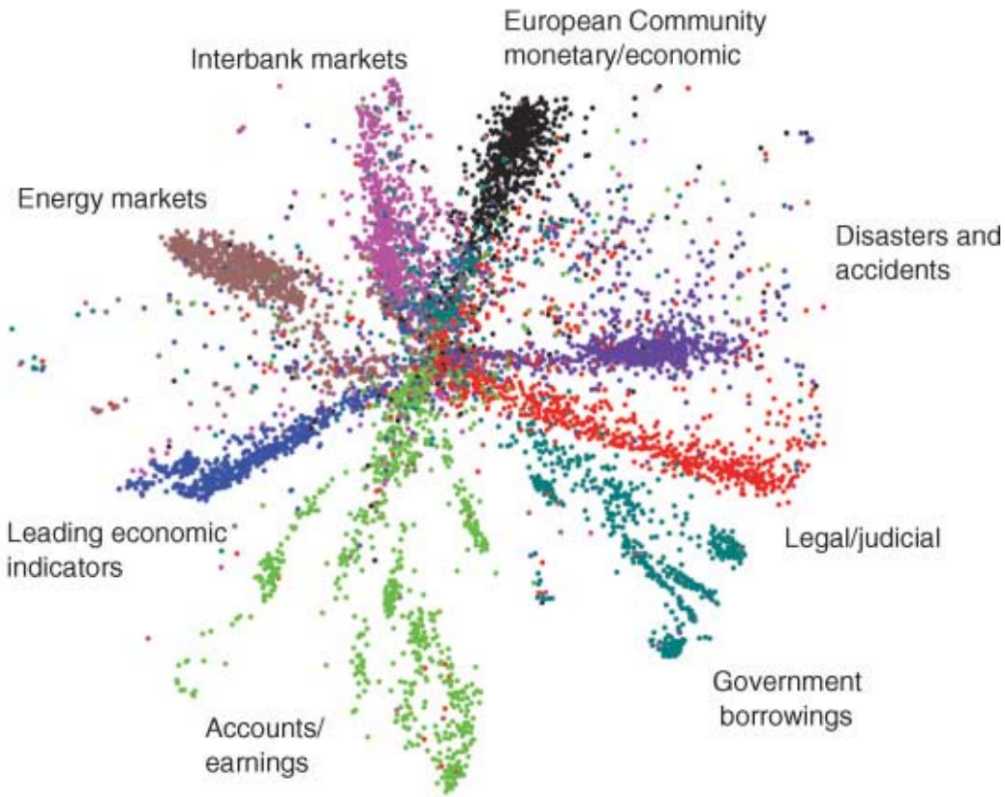
real data

30-D deep auto

30-D PCA

Autoencoder: Why are they useful?

Information retrieval: 804,414 newswire stories

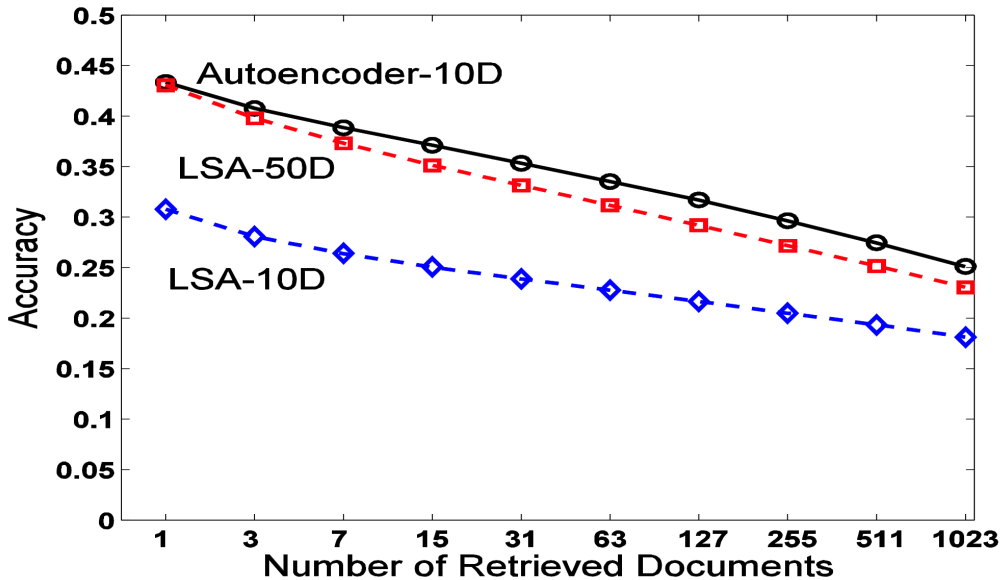


Autoencoder structure: 2000 — 500 — 250 — 125 — 2

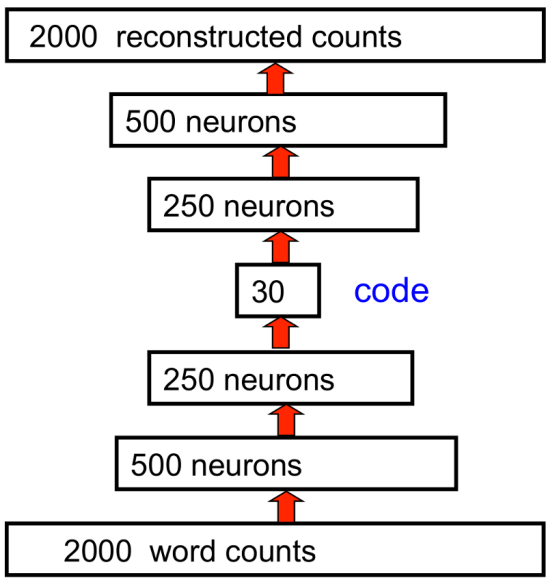
https://www.cs.toronto.edu/~hinton/science.pdf

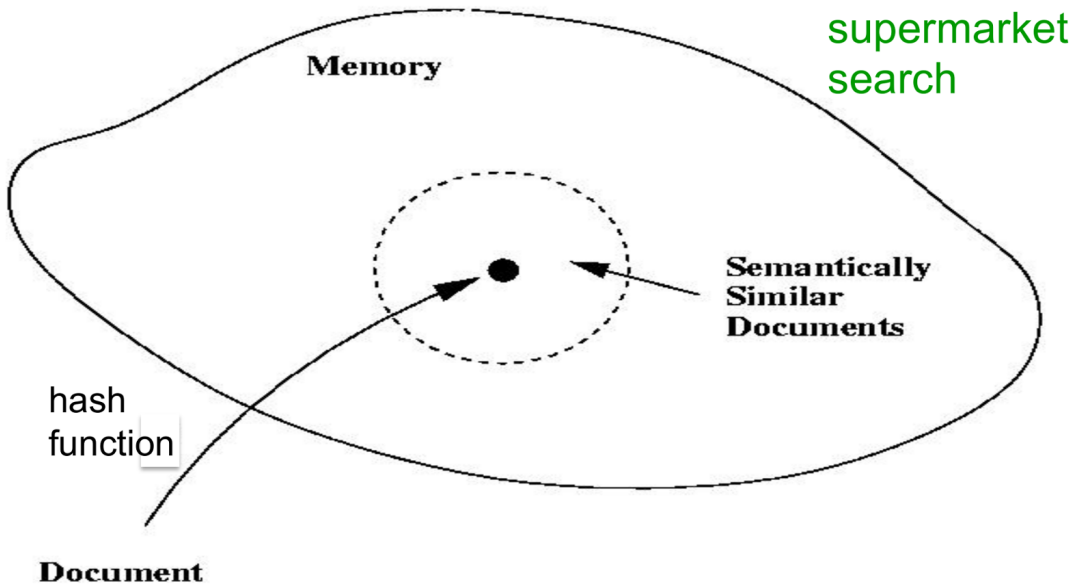
Comparison with PCA

Retrieval performance on 400,000 Reuters business news stories



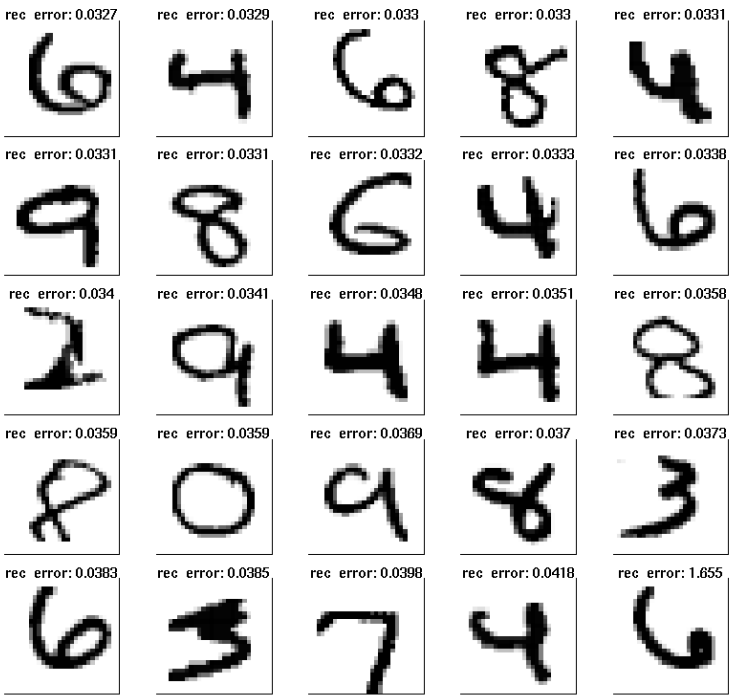
Finding binary codes for documents

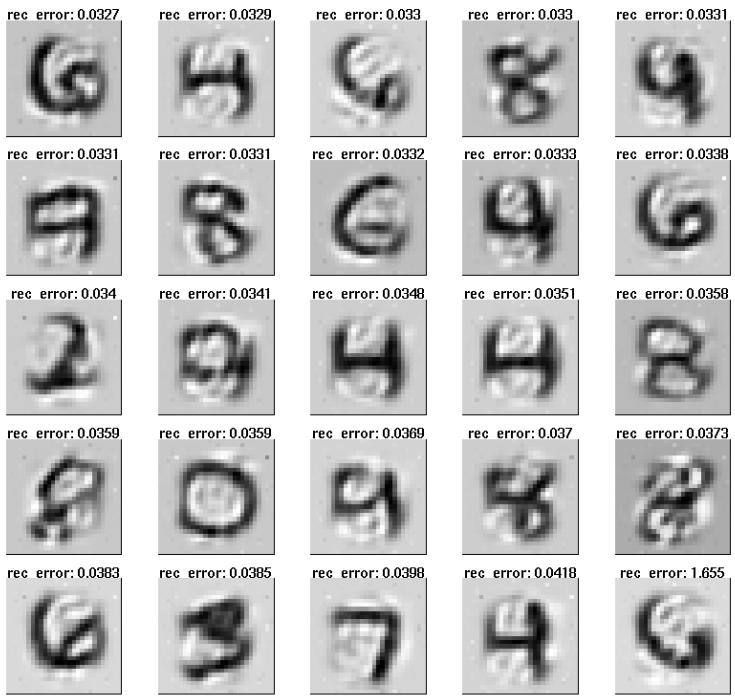




Autoencoder: Why are they useful?

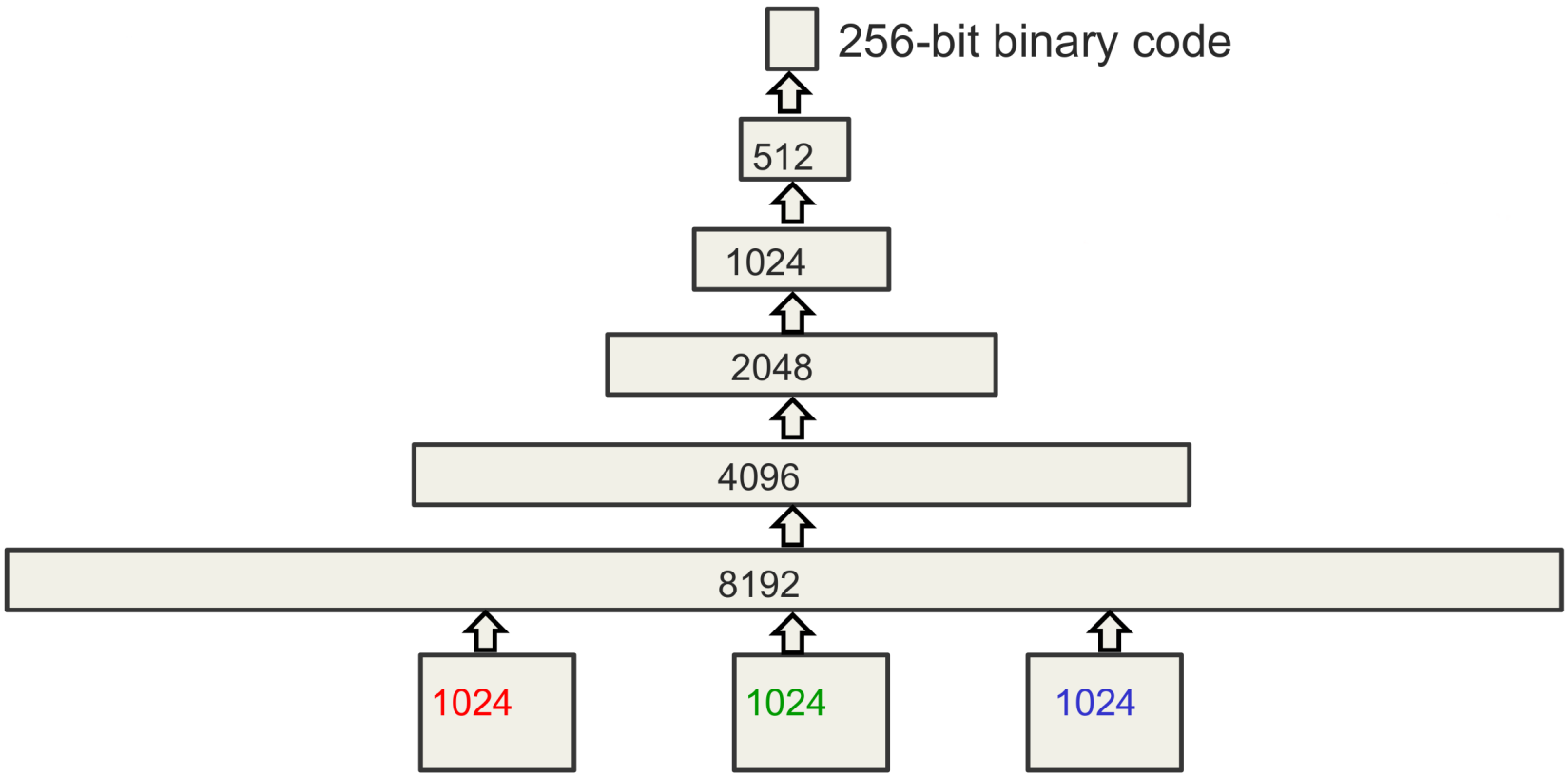
- unsupervised pretraining of weights (many unlabeled images, but only few labeled)
- anomaly detection



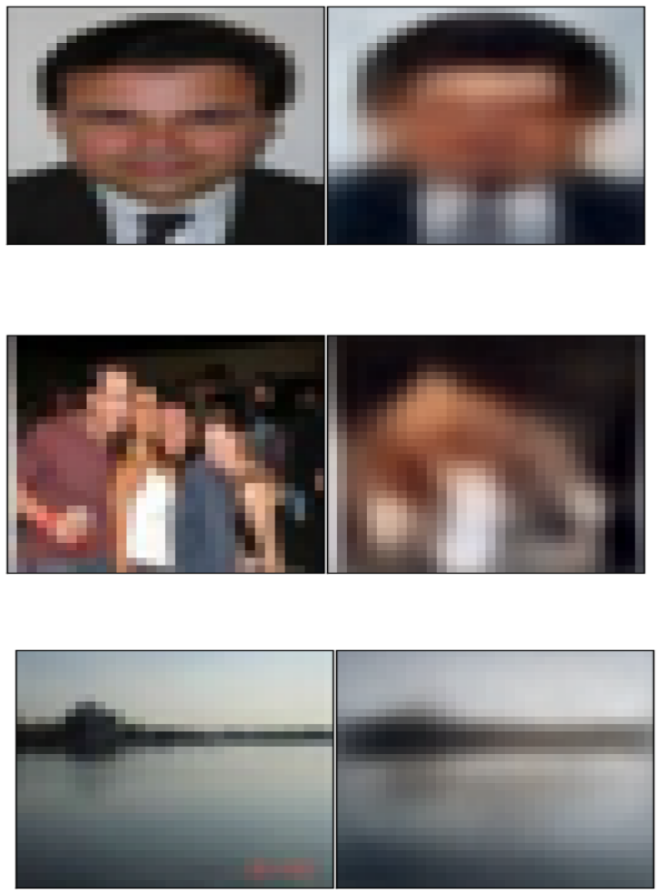


Autoencoder: Why are they useful?

Image retrieval



Reconstructions of 32x32 color images from 256-bit codes

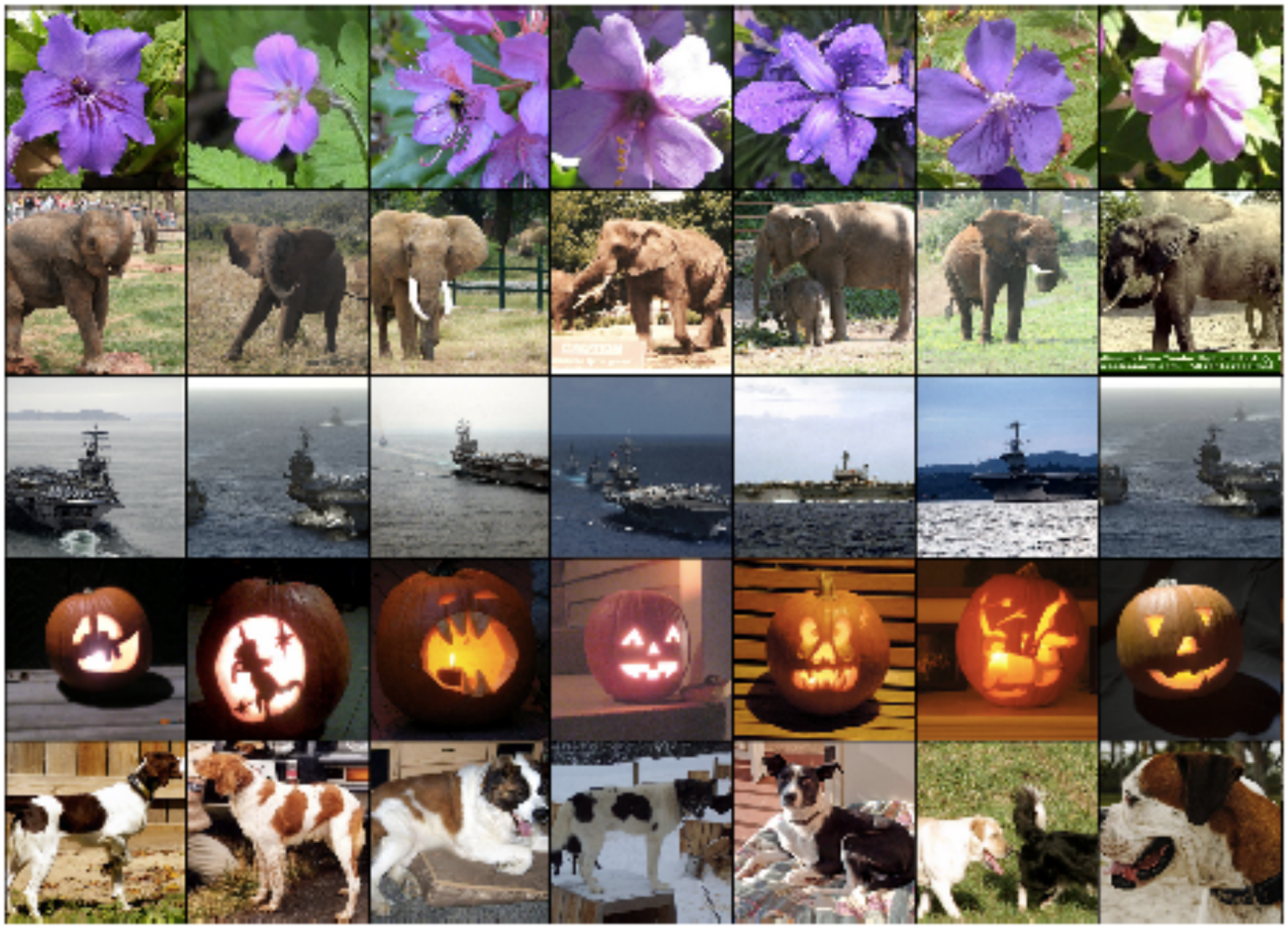


retrieved using 256 bit codes

retrieved using Euclidean distance in pixel intensity space

retrieved using 256 bit codes

retrieved using Euclidean distance in pixel intensity space

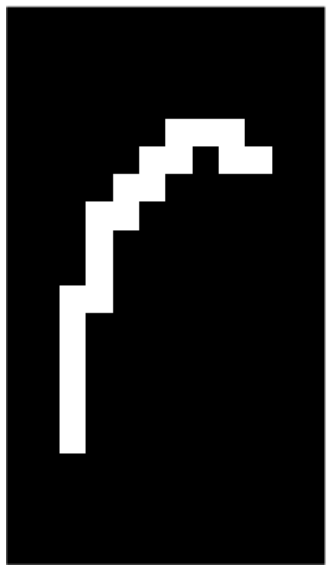


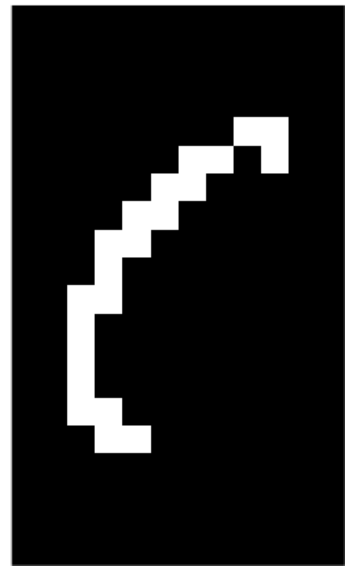
Leftmost column is the search image.

Other columns are the images that have the most similar feature activities in the last hidden layer.

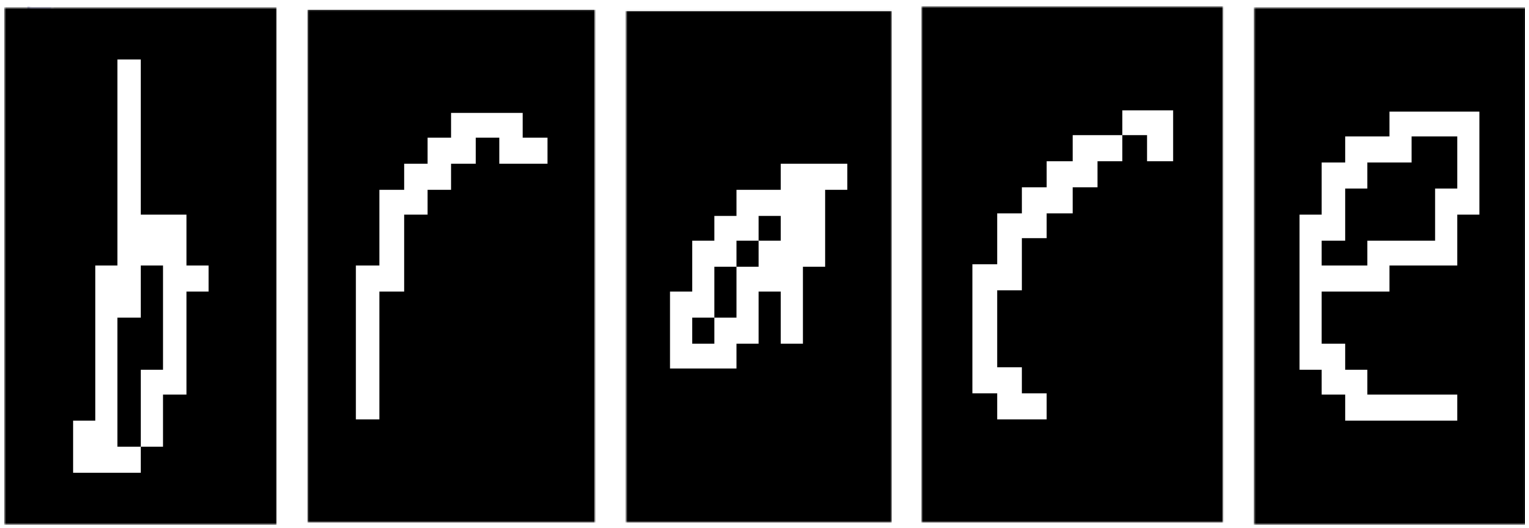
Sequence Models

Handwritten character recognition



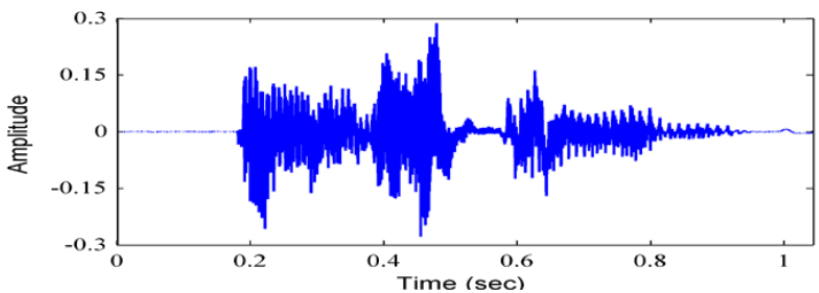


Structured prediction



Sequential data

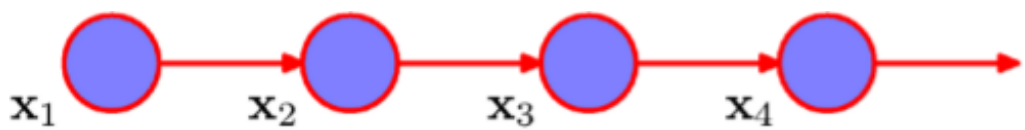
- time-series data (speach)
- characters in a sentence
- base pairs along a DNA strand



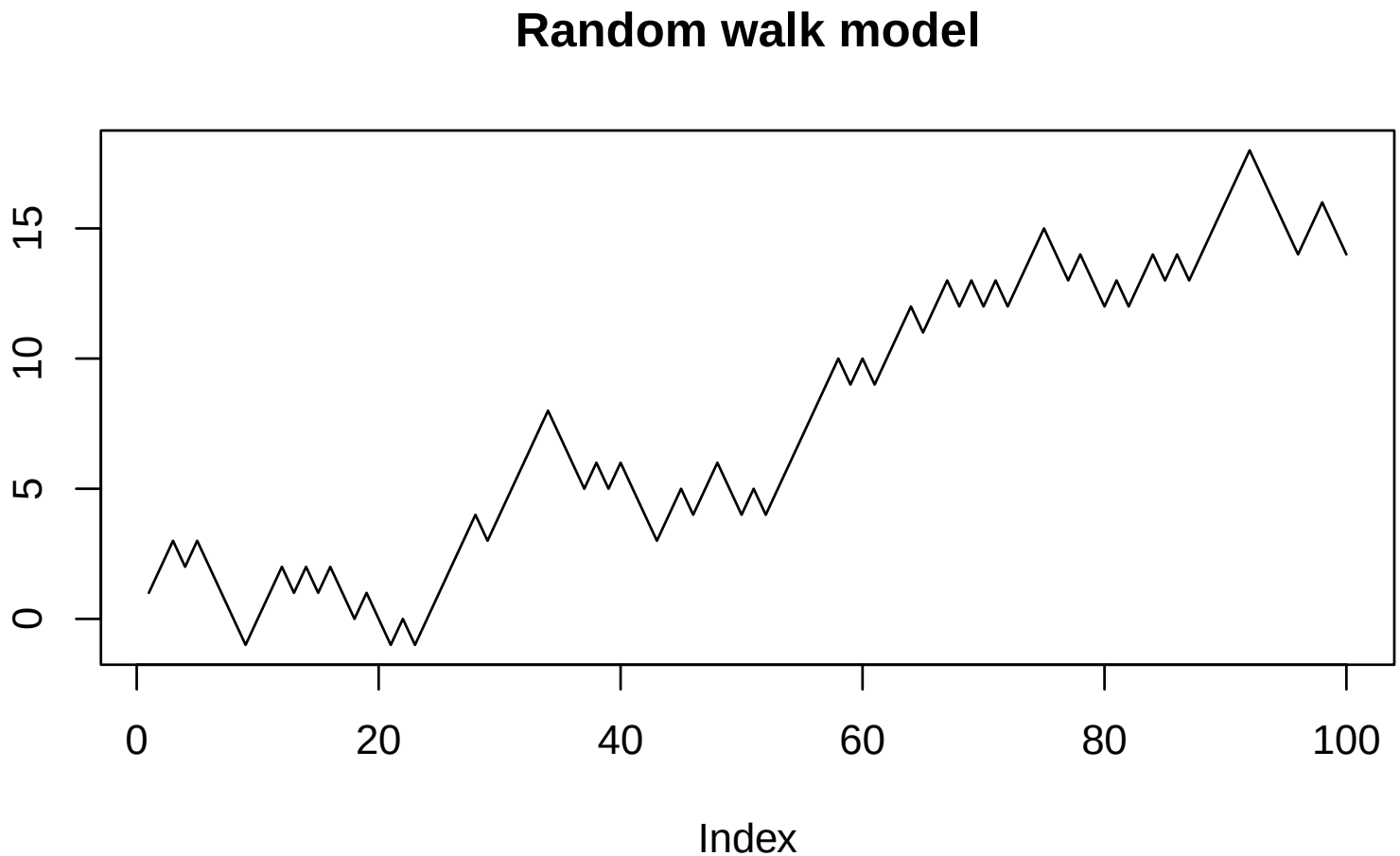
Markov Model

Joint distribution of n arbitrary random variables

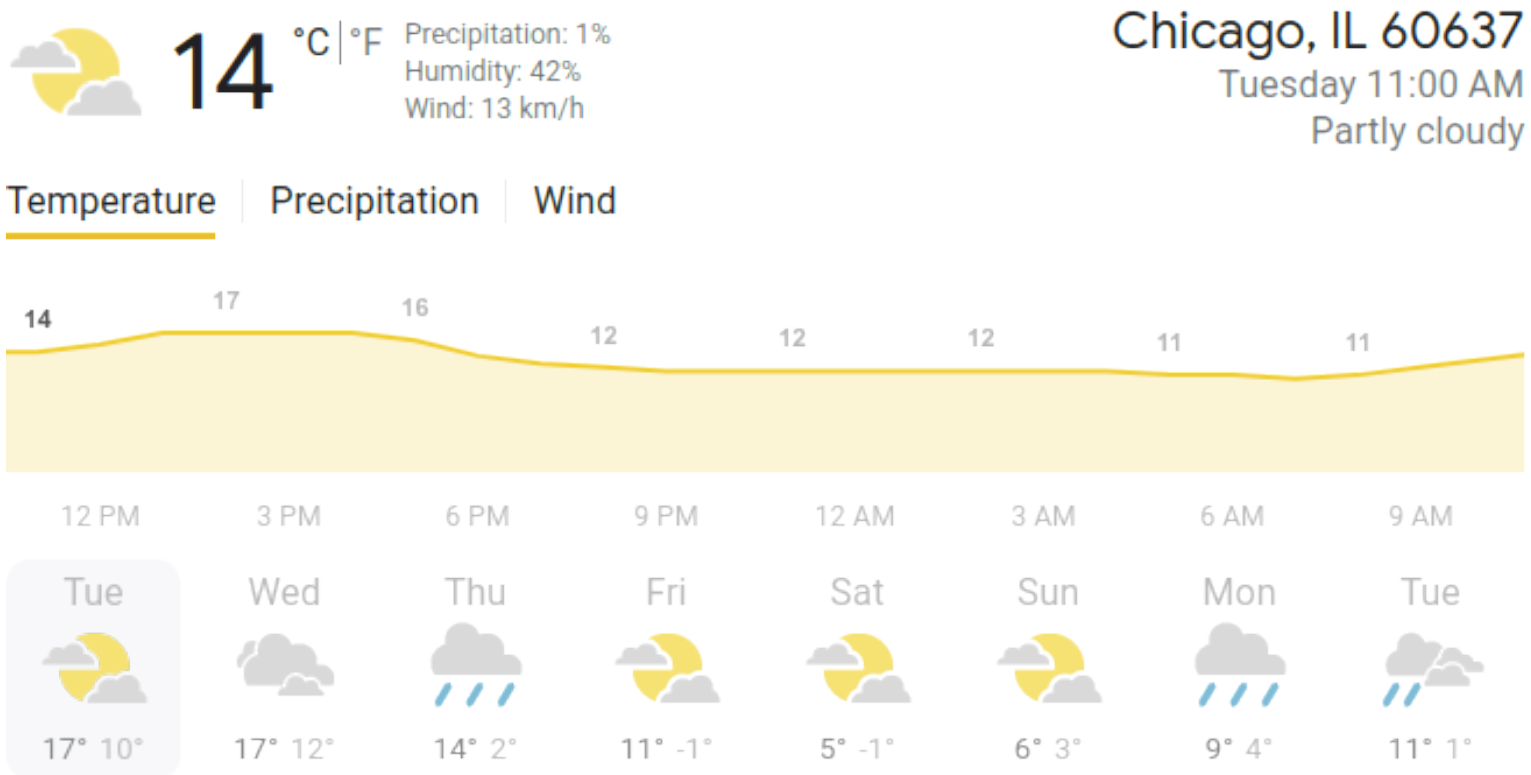
$$P(X_1, X_2, \ldots, X_n) = P(X_1) \cdot P(X_2 \mid X_1) \cdot \ldots \cdot P(X_n \mid X_{n-1})$$
$$= P(X_1) \cdot \prod_{i=2}^{n} P(X_i \mid X_{i-1})$$



Example



Example


14 °C | °F
Precipitation: 1%
Humidity: 42%
Wind: 13 km/h
Chicago, IL 60637
Tuesday 11:00 AM
Partly cloudy
Temperature
Precipitation
Wind
14
17
16
12
12
12
11
11
12 PM
3 PM
6 PM
9 PM
12 AM
3 AM
6 AM
9 AM
Tue 17° 10°
Wed 17° 12°
Thu 14° 2°
Fri 11° -1°
Sat 5° -1°
Sun 6° 3°
Mon 9° 4°
Tue 11° 1°

Recurrent neural networks

Recurrent neural networks

RNNs are very powerful, because they combine two properties:

- Distributed hidden state that allows them to store a lot of information about the past efficiently.
- Non-linear dynamics that allows them to update their hidden state in complicated ways.

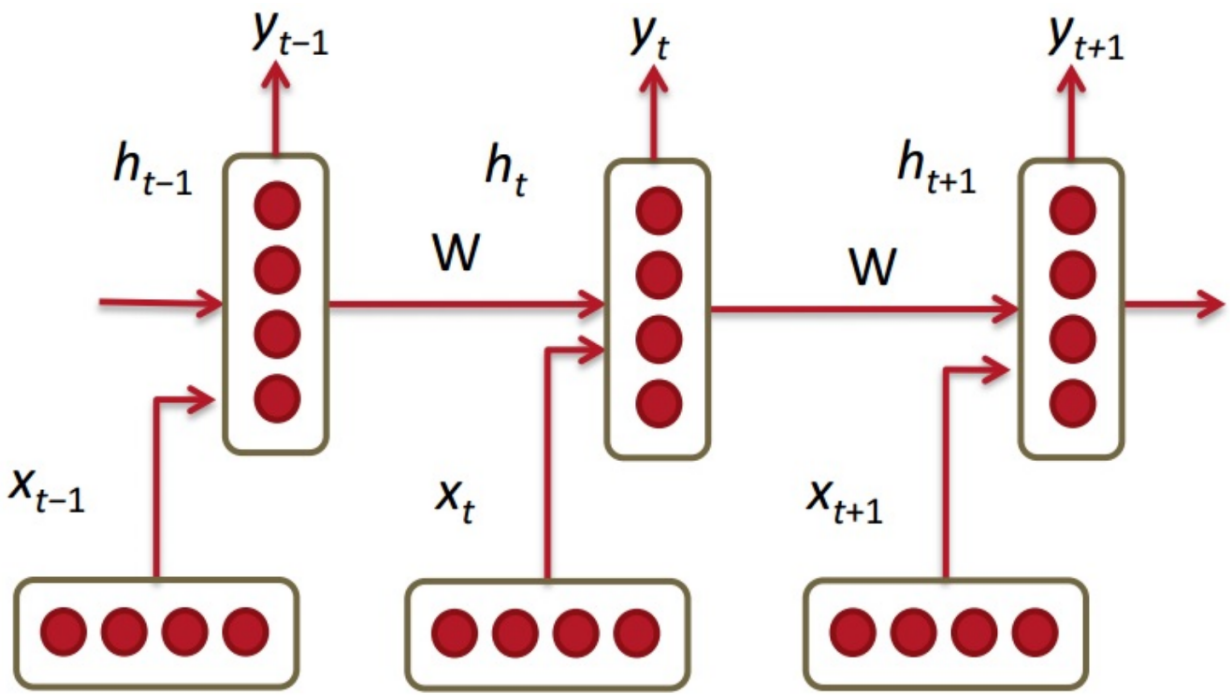
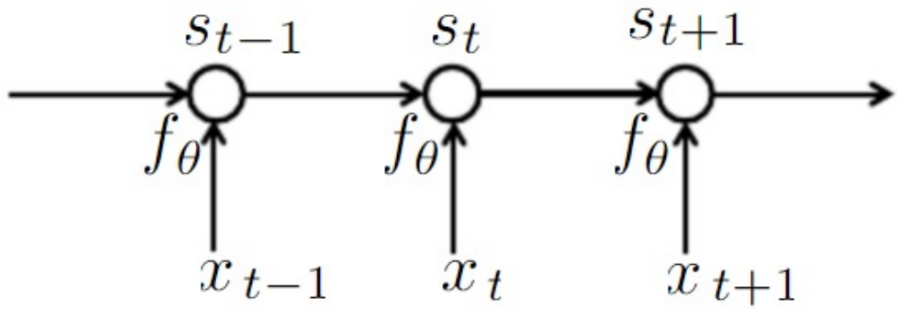


Figure: Richard Socher

Recurrent neural networks as Dynamic systems

$$s_t = f_\theta(s_{t-1}, x_t)$$



The state contains information about the whole past sequence.

$$s_t = g_t(x_t, x_{t-1}, x_{t-s}, \ldots, x_2, x_1)$$

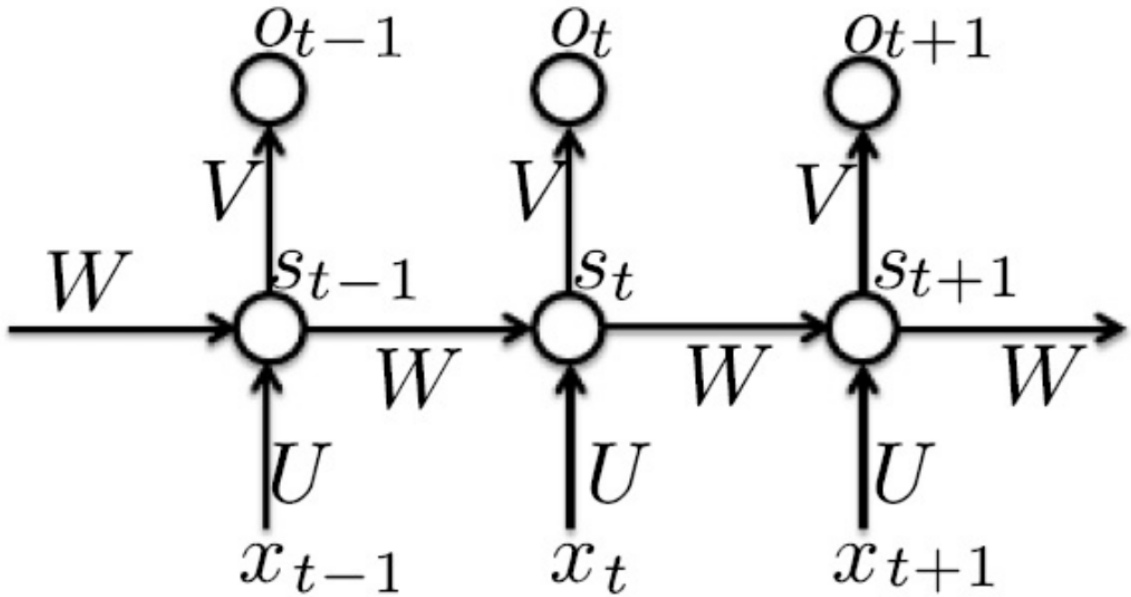
Parameter sharing

We can think of s_t as a summary of the past sequence of inputs up to t.

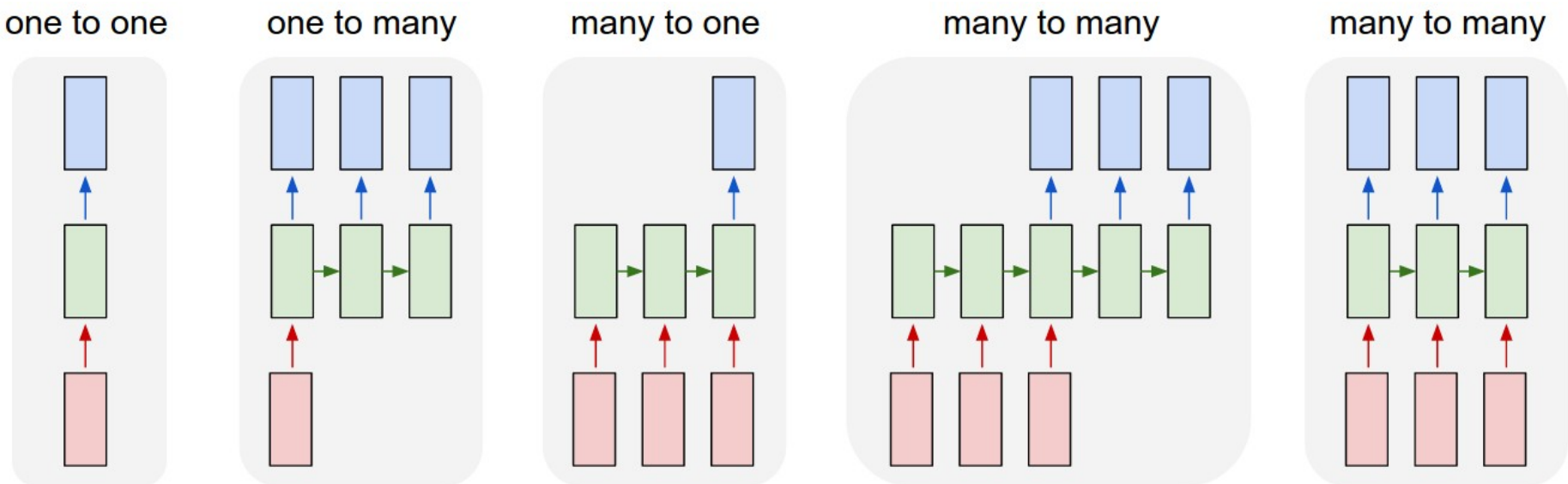
If we define a different function g_t for each possible sequence length, we would not get any generalization.

If the same parameters are used for any sequence length allowing much better generalization properties.

Recurrent Neural Networks



$$
\begin{aligned}
\mathbf{a}_t &= \mathbf{b} + W\mathbf{s}_{t-1} + U\mathbf{x}_t \\
\mathbf{s}_t &= tanh(\mathbf{a}_t) \\
\mathbf{o}_t &= \mathbf{c} + V\mathbf{s}_t \\
\mathbf{p}_t &= softmax(\mathbf{o}_t)
\end{aligned}
$$


one to one
one to many
many to one
many to many
many to many

Recurrent Neural Networks: Process Sequences

One to one

- vanilla neural network

One to many

- image captioning (image –> sequence of words)

Many to one

- sentiment classification (sequence of words –> sentiment)

Many to many

- machine translation (sequence of words –> sequence of words)
- video classification on frame level

Example: Sentiment Classification

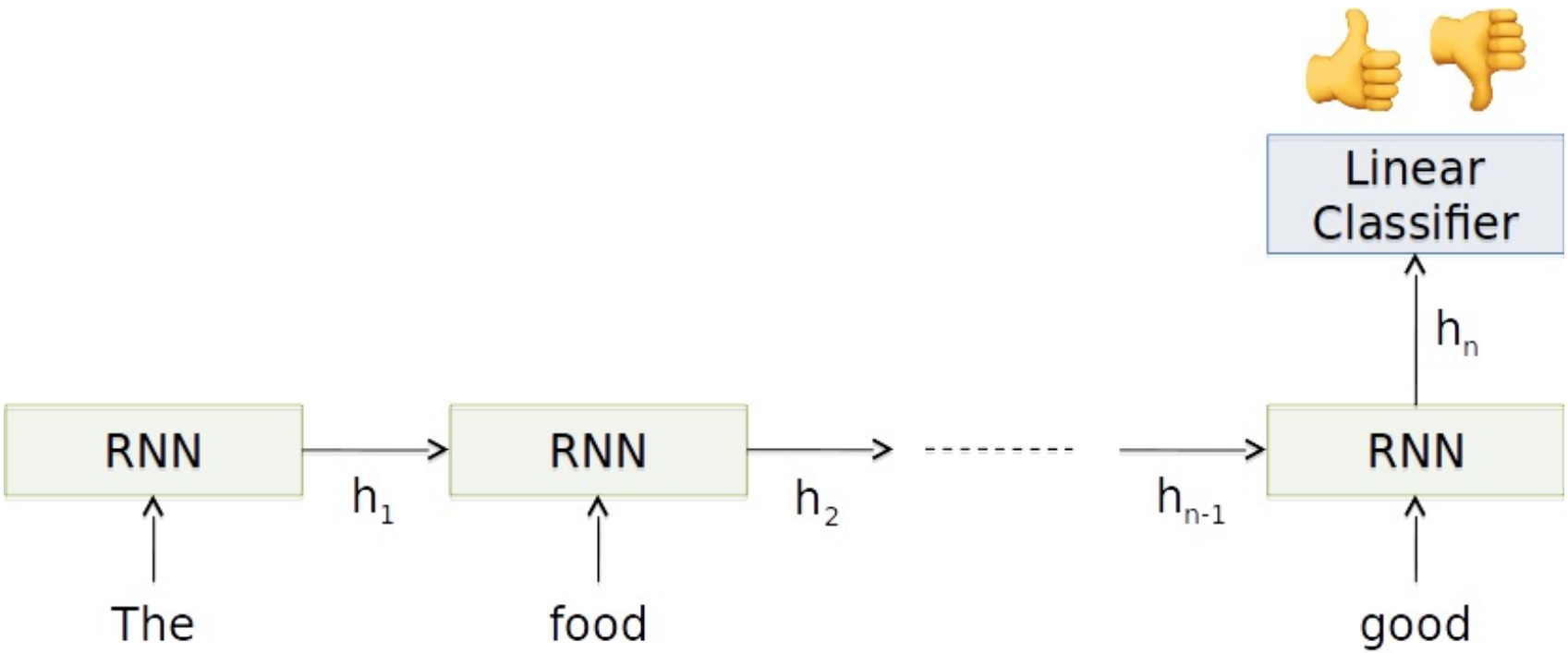
Classify a restaurant review from Yelp! OR movie review from IMDB as positive or negative

Inputs: Multiple words, one or more sentences

Outputs: Positive / Negative classification

"The food was really good"

Example: Sentiment Classification



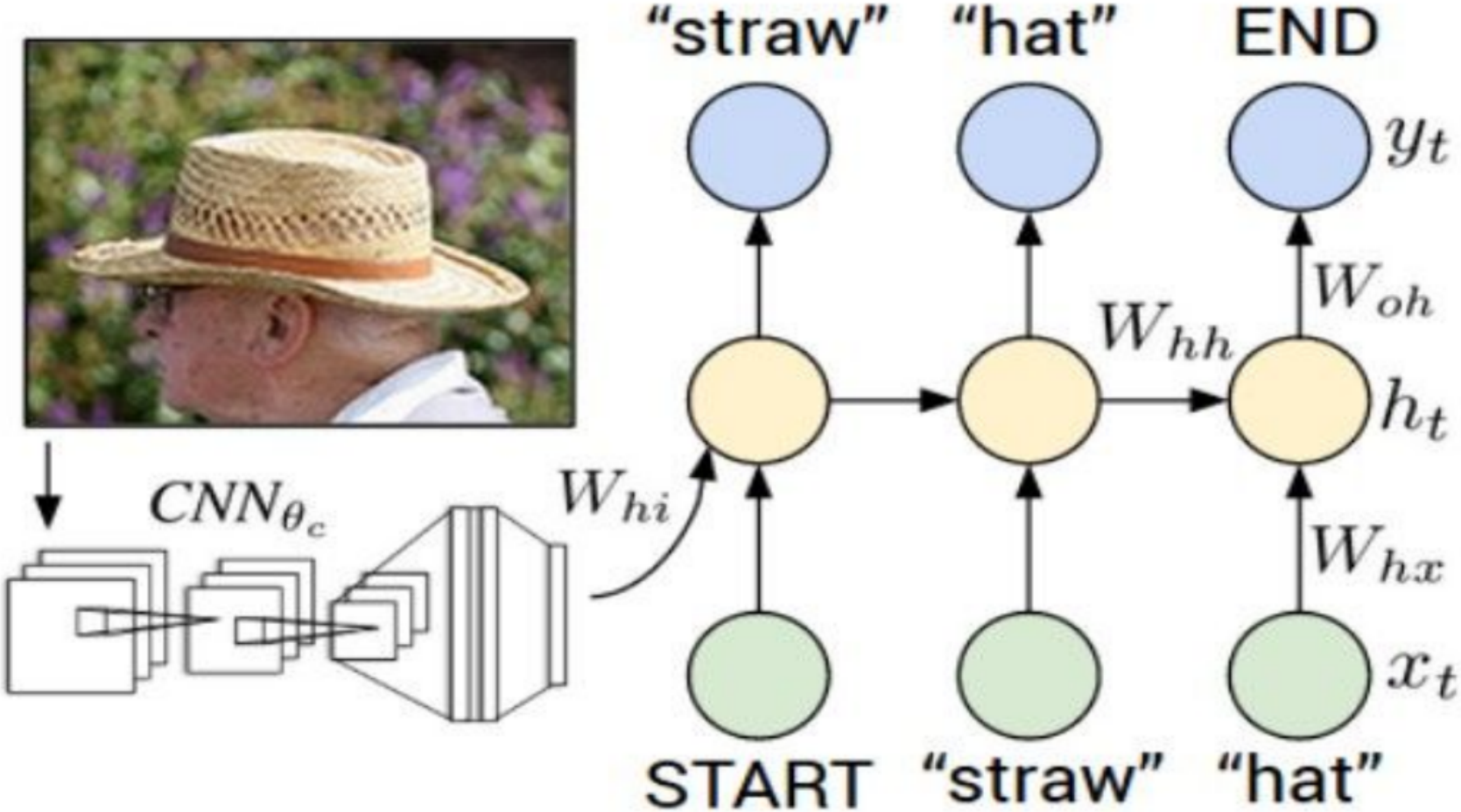
Example: Image Captioning

Given an image, produce a sentence describing its contents

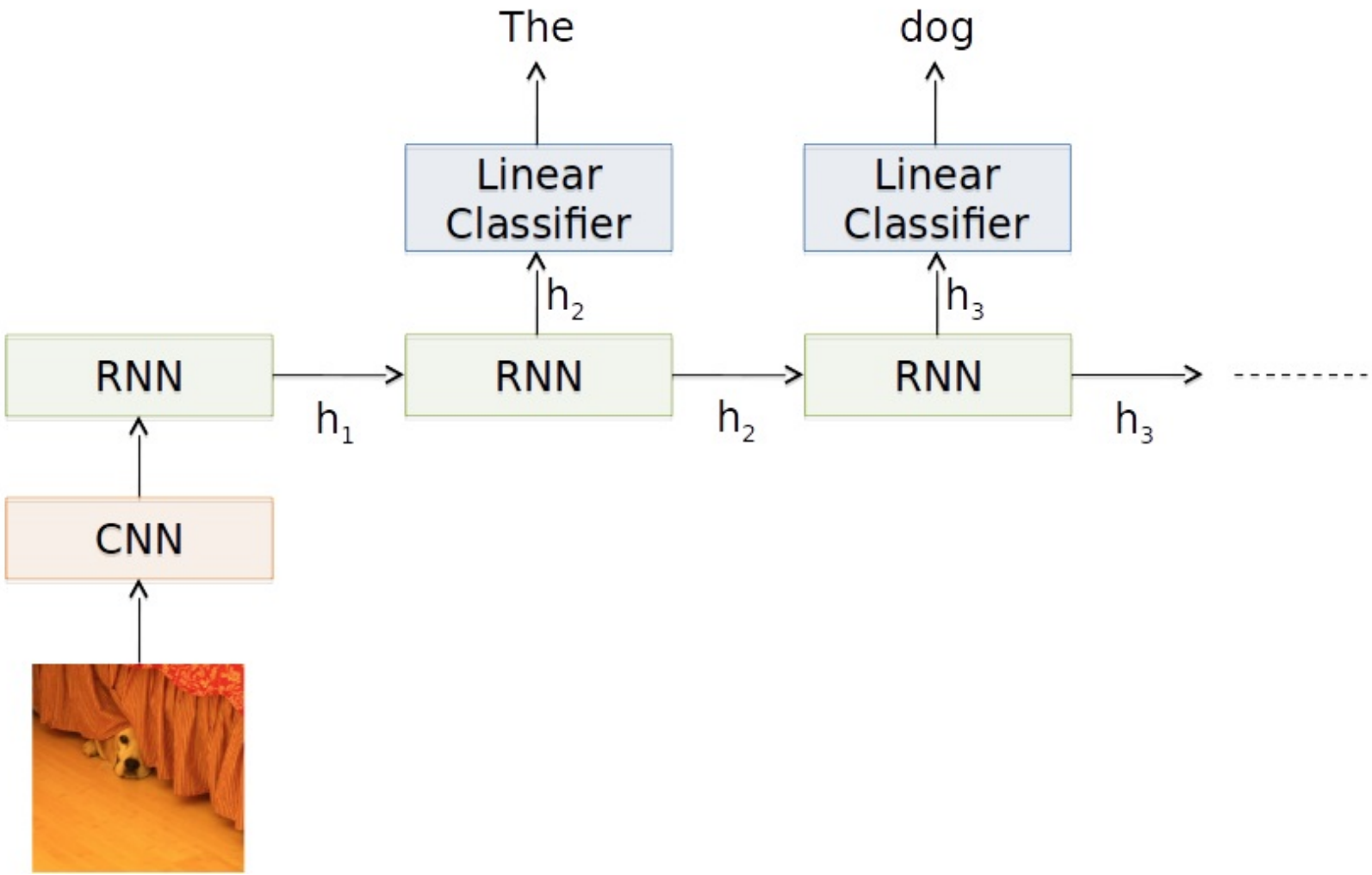
Inputs: Image feature (from a convolutional neural network)

Outputs: Multiple words (one sentence)

Example: Image Captioning



Example: Image Captioning



Example: Image Captioning

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

A man riding a dirt bike on a dirt track

Example: Image Captioning

A woman is holding a cat in her hand

A person holding a computer mouse on a desk

A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

Examples

Handwritting
`https://www.cs.toronto.edu/~graves/handwriting.html`

Image captioning
`https://cocodataset.org/#explore`