ICS 691D: DEEP LEARNING REVIEW

Peter Y. Washington, PhD
Assistant Professor, ICS
August 29, 2022

OUTLINE

- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

OUTLINE

- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

UPCOMING ASSIGNMENTS DUE

- Wednesday August 31: Proposal topic
- Wednesday September 7: Haber reflection
- Monday September 12: Akalin reflection

PROPOSAL TOPIC SUBMISSION

- Choose a topic from the course calendar that interests you based on the overview from last class
- Provide a paragraph-long summary describing your project proposal
- Submit on Laulima
- Requested format:
  - *Project Category: [choose from course calendar]*
  - *Societal and/or Technical Motivation: [text]*
  - *Dataset: [include size, data description, and whether it is a publicly available dataset or how the data will be collected]*
  - *Methods: [a few sentences about the proposed methodology]*

PROPOSAL TOPIC SUBMISSION

- Choose a topic from the course calendar that interests you based on the overview from last class
- Provide a paragraph-long summary describing your project proposal
- Submit on Laulima
- Requested format:
  - *Project Category: [choose from course calendar]*
  - *Societal and/or Technical Motivation: [text]*
  - *Dataset: [include size, data description, and whether it is a publicly available dataset or how the data will be collected]*
  - *Methods: [a few sentences about the proposed methodology]*

Questions about your planned topic? Class time to discuss.

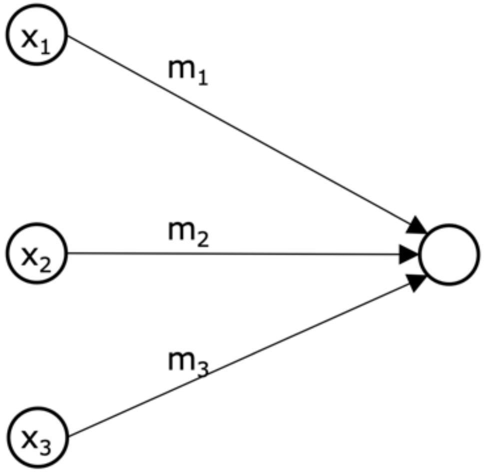
REMINDER ABOUT TODAY'S CLASS

- We will cover more math today than usual.
- For those who have taken ML and DL courses, this will be review.
- For those who are newer to ML, you will still get something out of the big picture.

OUTLINE

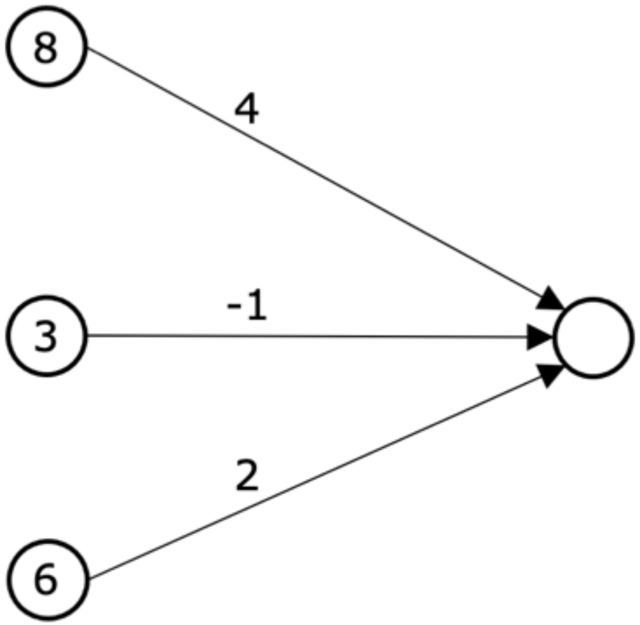
- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

THE PERCEPTRON

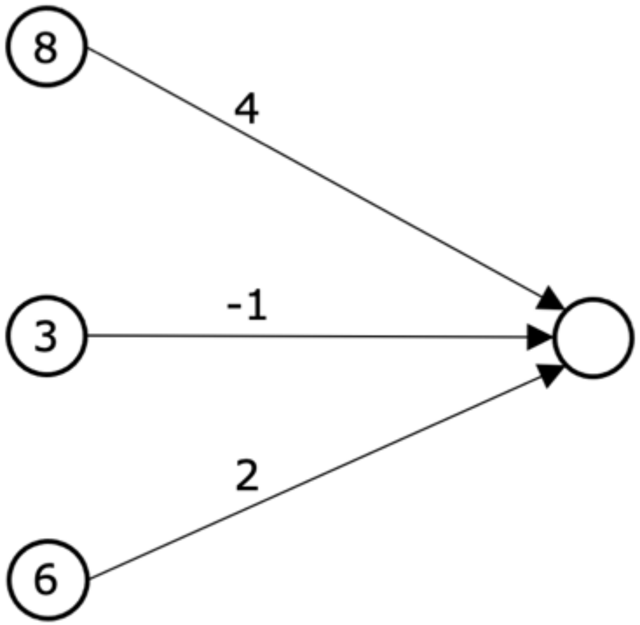


$$y(X) = \begin{cases} 1 \; if \; \sum_{i=1}^{N} m_i x_i + b > 0 \\ 0 \; otherwise \end{cases}$$

WHAT IS THE VALUE OF THE RIGHT-MOST NODE?

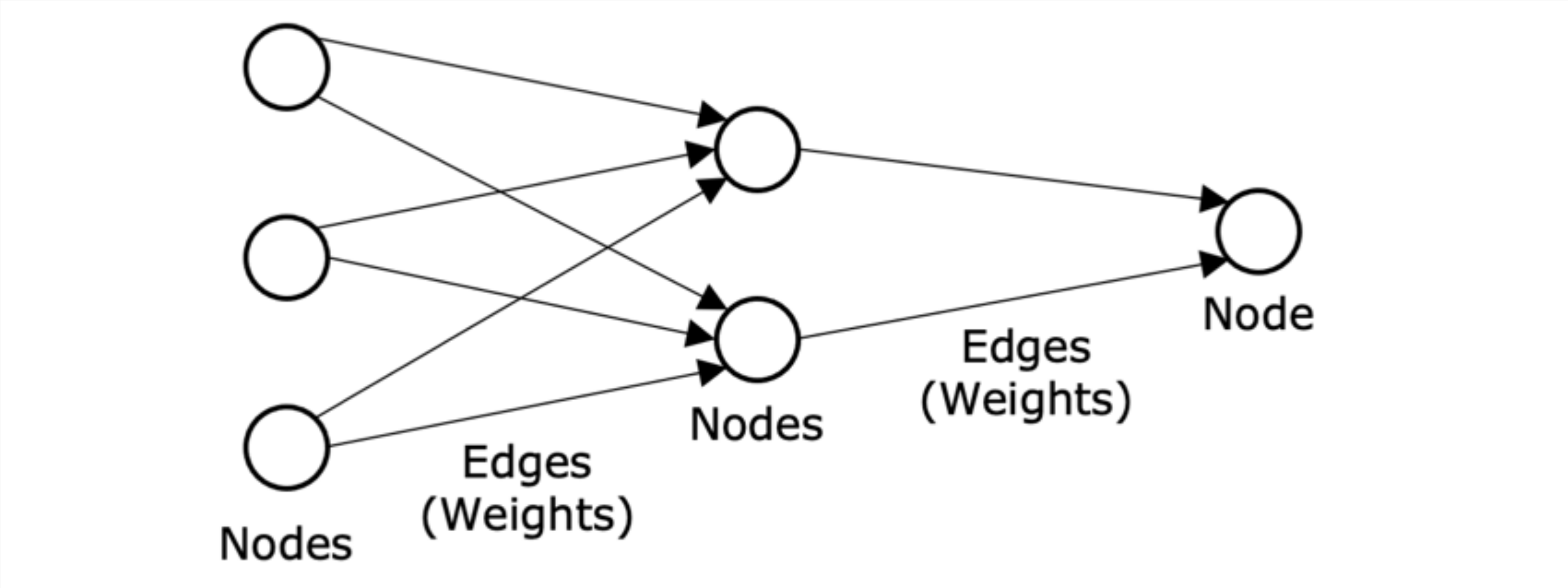


WHAT IS THE VALUE OF THE RIGHT-MOST NODE?

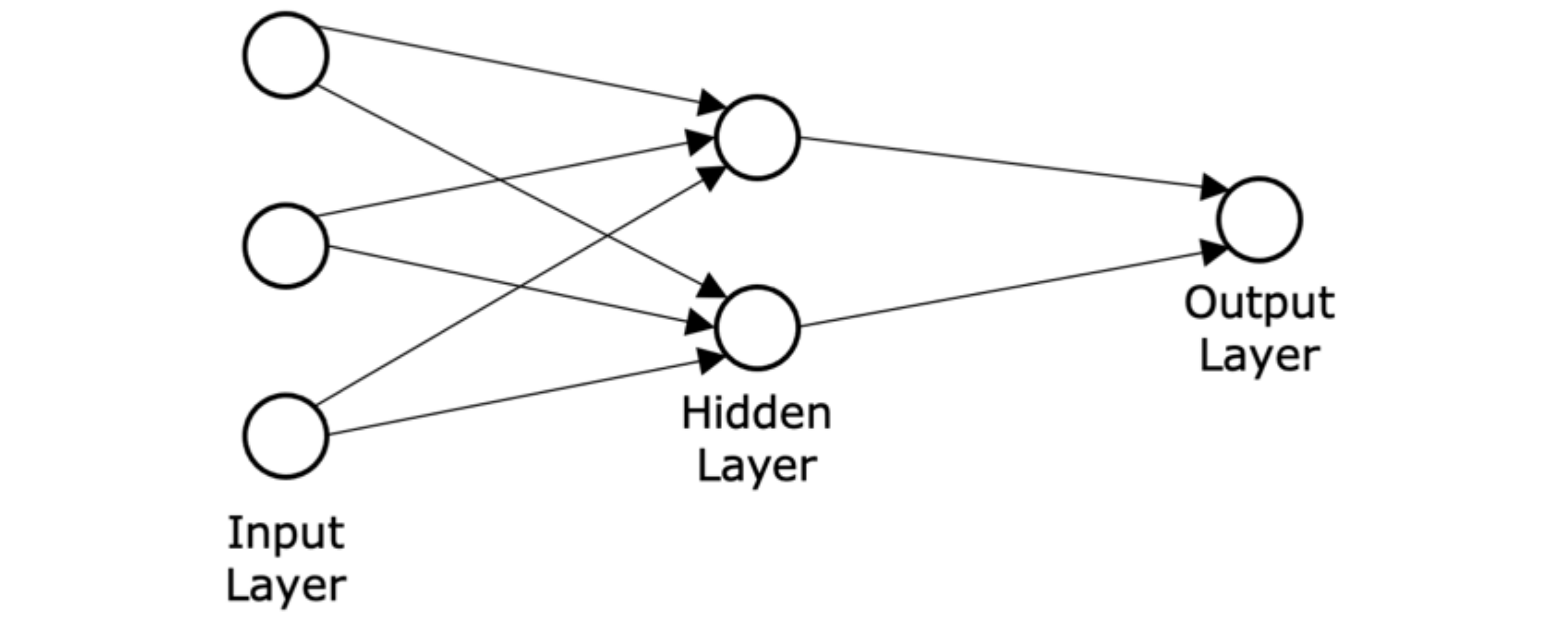


$$8 \times -4 + 3 \times -1 + 6 \times 2 = -23$$

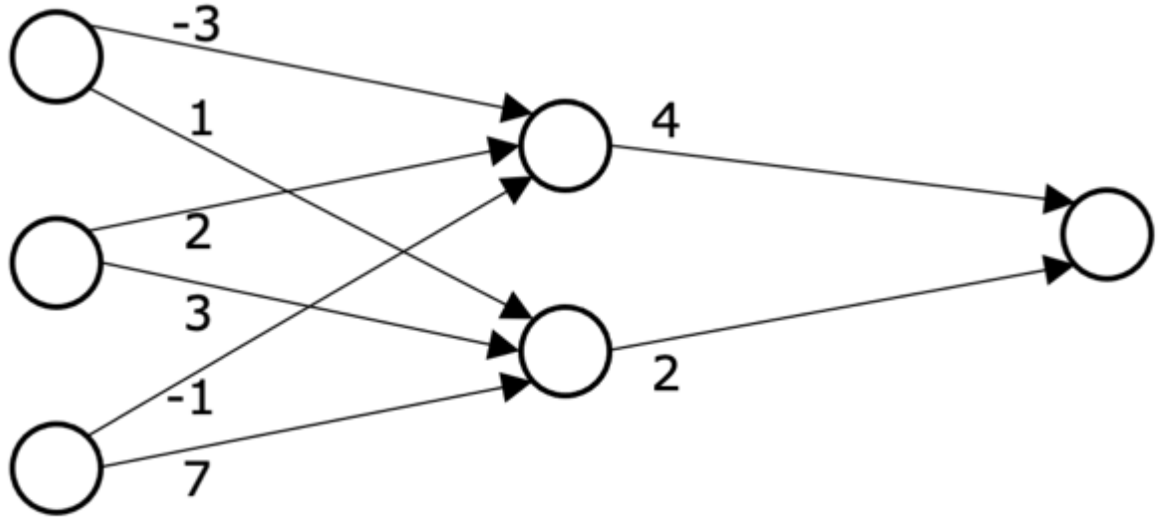
FEEDFORWARD NEURAL NETWORKS



FEEDFORWARD NEURAL NETWORKS



FEEDFORWARD NEURAL NETWORKS



PRIMARY NEURAL NETWORK ARCHETYPES

- **Dense neural networks:** tabular data
- **Convolutional neural networks (CNNs):** image data (or spatial data more broadly)
- **Recurrent neural networks (RNNs):** time series data
- **Transformers:** time series data

BUT WAIT…

Aren’t all neural networks just linear functions? Can’t we just use linear or logistic regression?

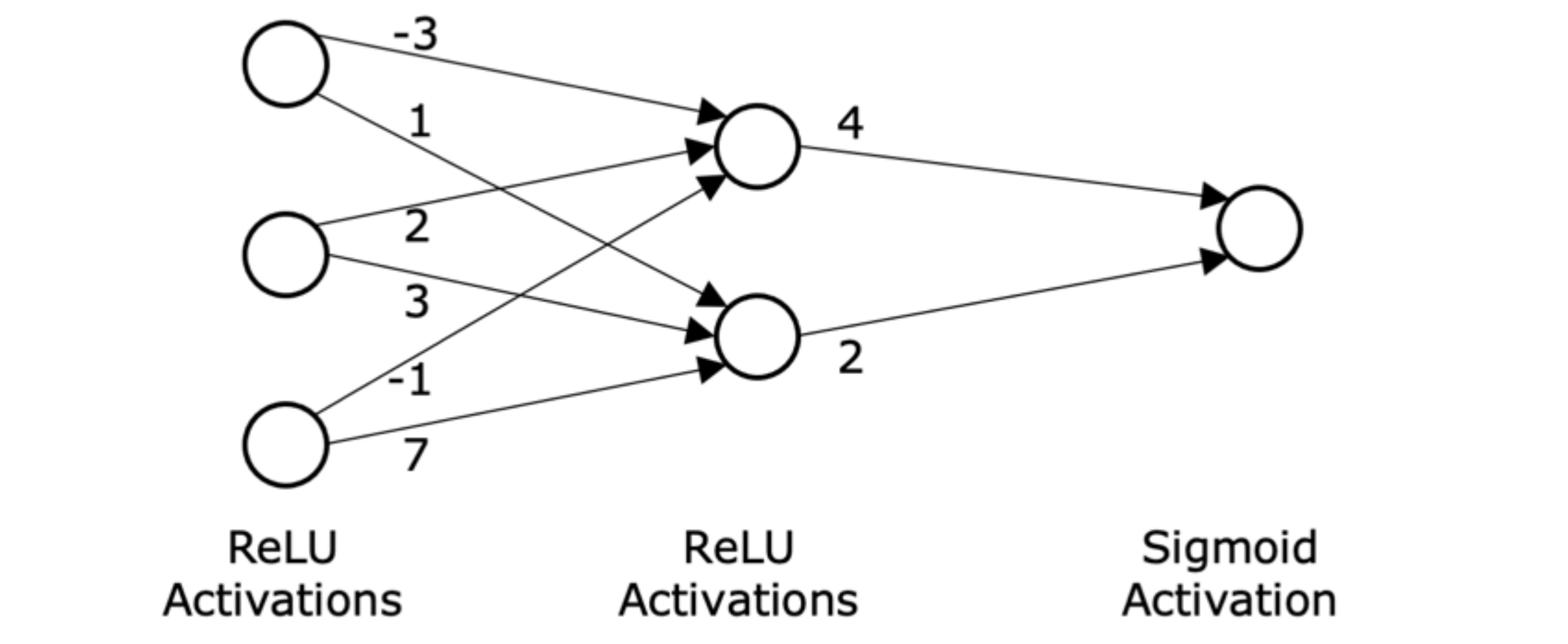
REVISITING ACTIVATION FUNCTIONS

$$\text{Probability}(\text{Y} = 1) = \text{Sigmoid Activation}(\text{mx} + \text{b}) = \frac{1}{1+e^{-mx+b}}$$

RELU ACTIVATION

$$ReLU(x) = \begin{cases} x \; if \; x > 0 \\ 0 \; if \; x \leq 0 \end{cases} = \max(0, x)$$

DENSE NEURAL NETWORK, AKA FULLY CONNECTED NETWORK



EXAMPLE OF DISCRIMINATION FUNCTION THAT CAN BE LEARNED WITH DEEP LEARNING USING NONLINEAR ACTIVATIONS

F(a,b,c) =

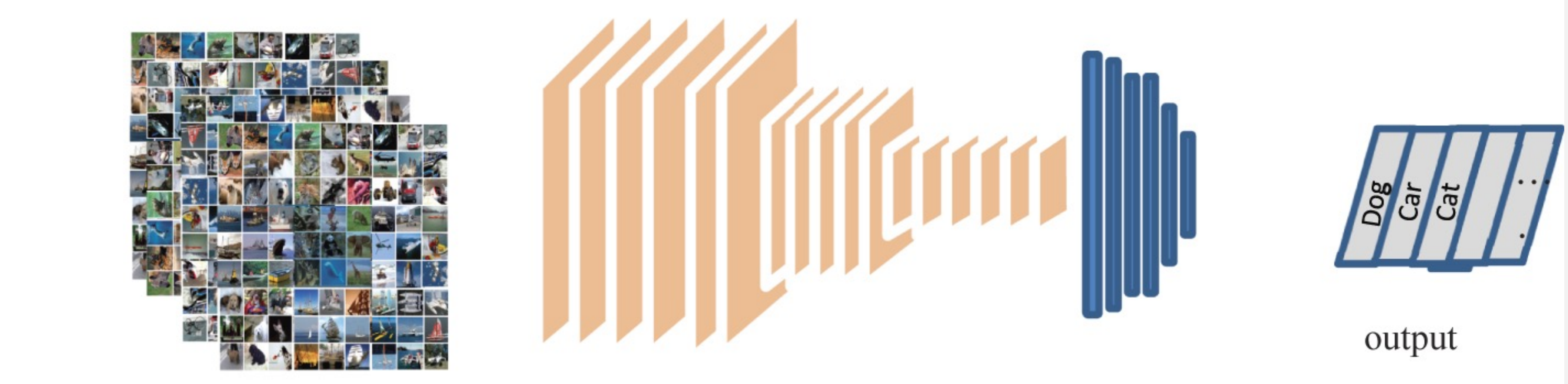
$$\begin{cases} \sqrt[3]{\iiint_{3.22}^{6.11} \sqrt{a^2 + b^2 - e^{-ib}}}, a < 5b \; and \; \tanh^{-1} b \succ \max_{0 \le c \le 1} 4.11abe^{-c^2} \\ \begin{vmatrix} \frac{\partial a \sec bc}{\partial b} & \begin{pmatrix} c \\ ab - \frac{\pi}{b} \end{pmatrix} \\ \sum_c^a ba - c & b^{44.45} \end{vmatrix} - 1, c < 0 \; and \; a > 0 \; and \; condition \; 1 \; not \; met \\ \dots \end{cases}$$

BACKPROPAGATION

$$\frac{\partial L}{\partial ab} = \frac{\partial L}{\partial Z}\frac{\partial Z}{\partial Y}\frac{\partial Y}{\partial X}\frac{\partial X}{\partial W}\frac{\partial W}{\partial V}\cdots\frac{\partial D}{\partial C}\frac{\partial C}{\partial B}\frac{\partial B}{\partial ab}$$

WHAT IS DEEP LEARNING?

Neural nets with many layers!

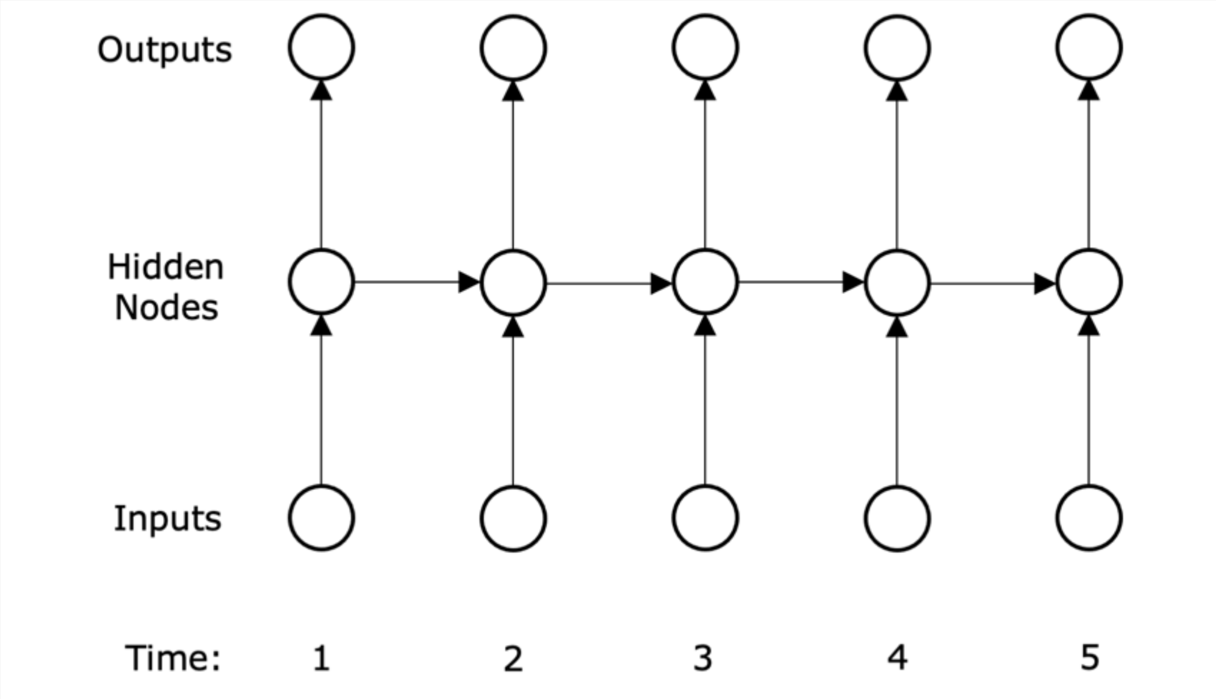


https://www.emerald.com/insight/content/doi/10.1108/ACI-07-2021-0191/full/html

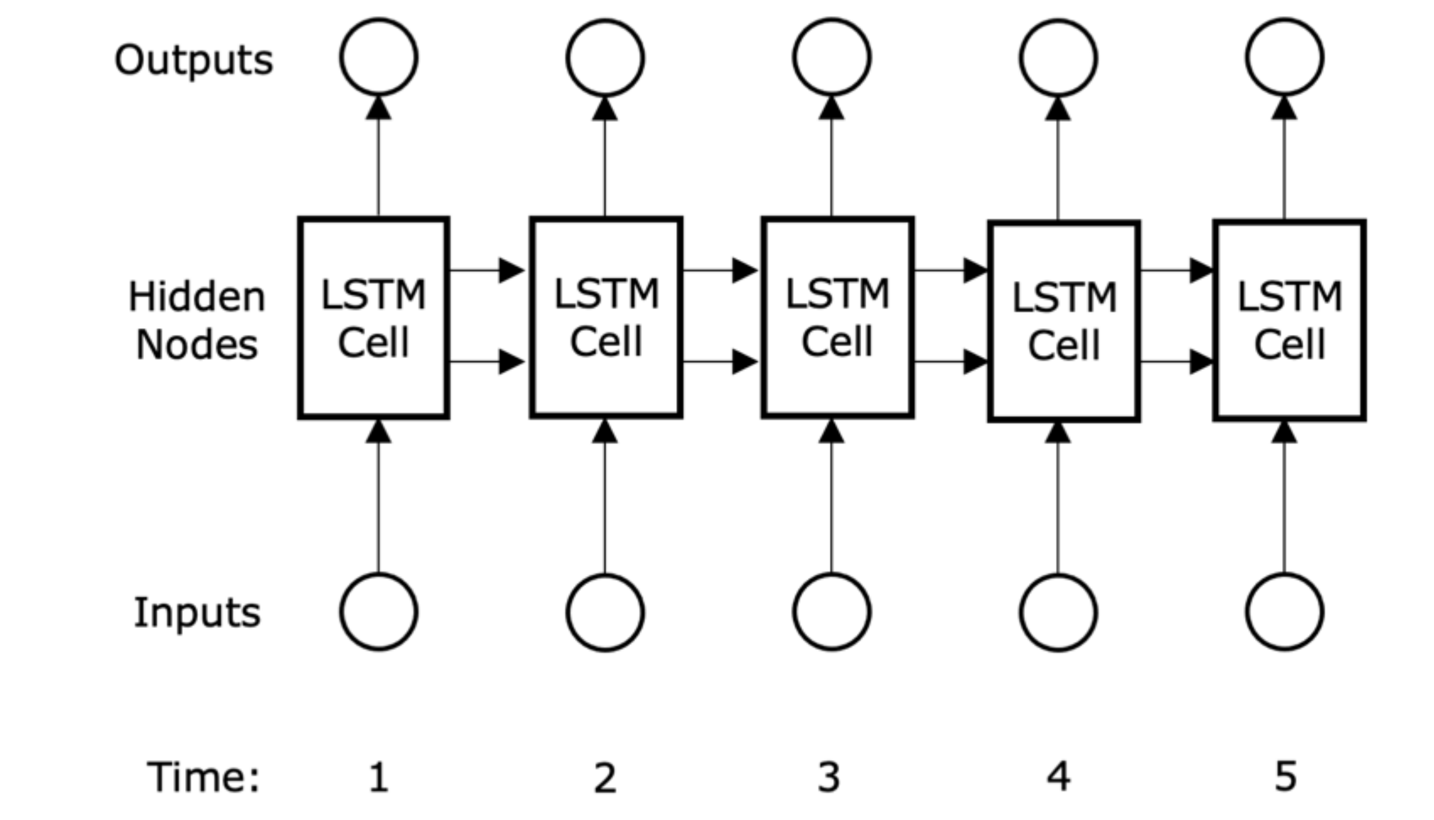
DEMO: TENSORFLOW PLAYGROUND

https://playground.tensorflow.org/

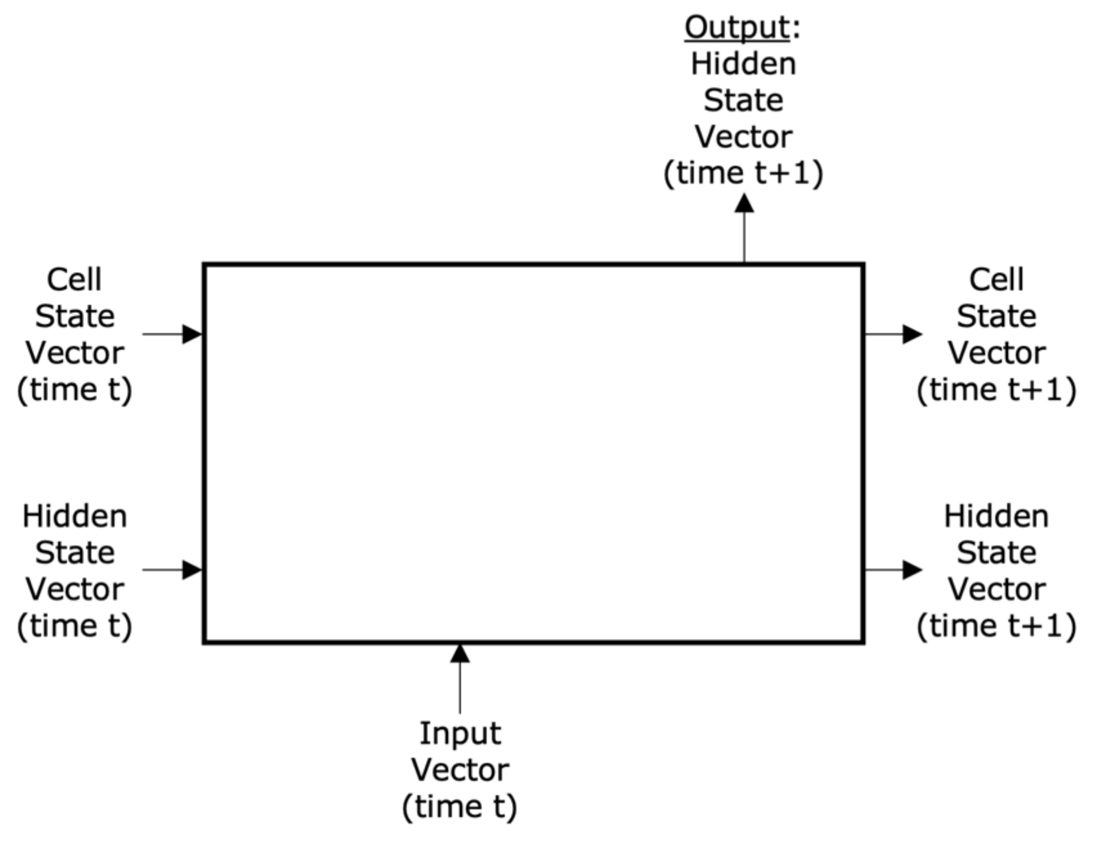
RECURRENT NEURAL NETWORKS



LONG SHORT-TERM MEMORY NETWORKS (LSTMS)

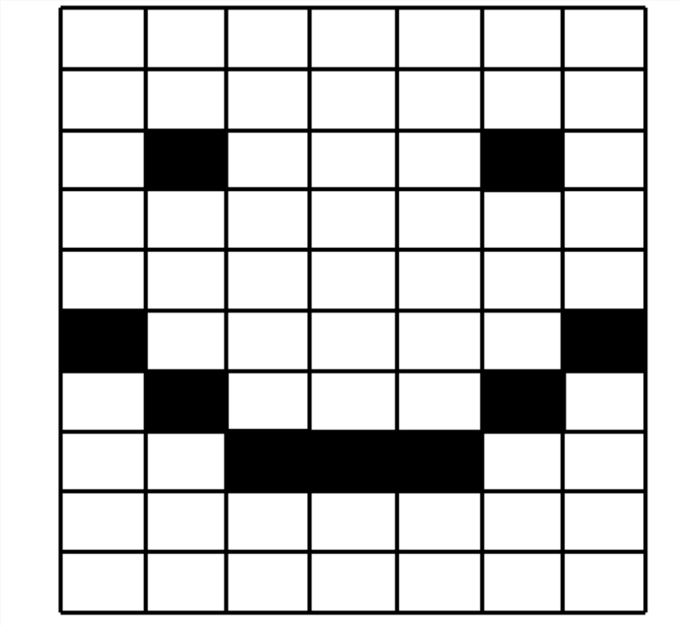


LONG SHORT-TERM MEMORY NETWORKS (LSTMS)



CONVOLUTIONAL NEURAL NETWORKS (CNNS)

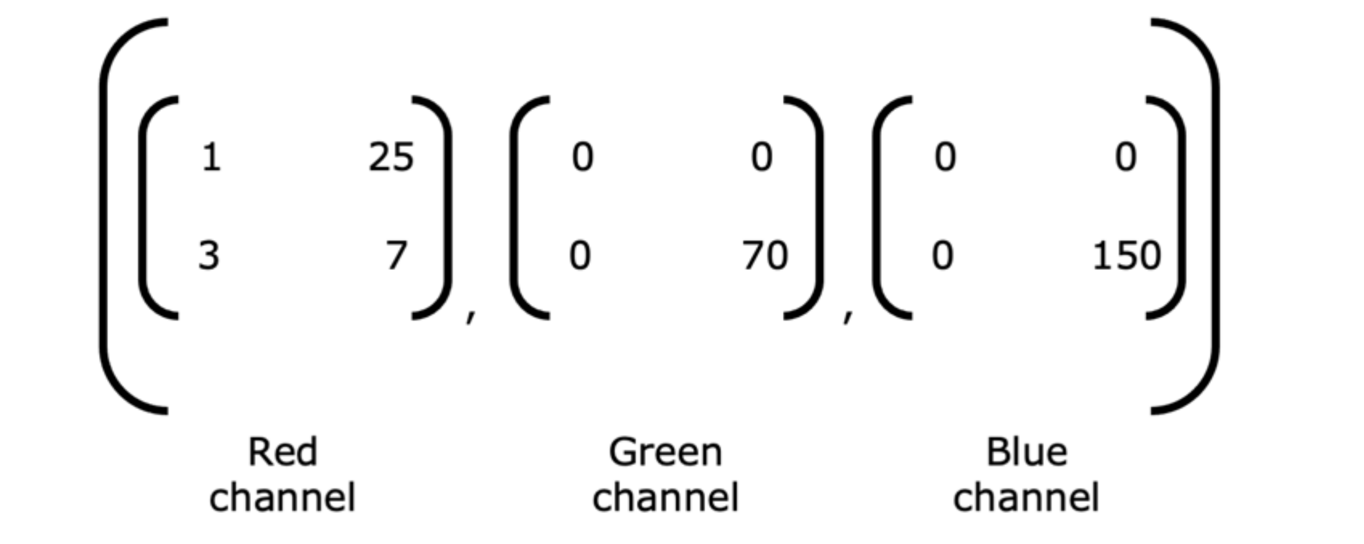
Image representation on a computer:



$$
\begin{pmatrix}
255 & 255 & 255 & 255 & 255 & 255 & 255 \\
255 & 255 & 255 & 255 & 255 & 255 & 255 \\
255 & 0 & 255 & 255 & 255 & 0 & 255 \\
255 & 255 & 255 & 255 & 255 & 255 & 255 \\
255 & 255 & 255 & 255 & 255 & 255 & 255 \\
0 & 255 & 255 & 255 & 255 & 255 & 0 \\
255 & 0 & 255 & 255 & 255 & 0 & 255 \\
255 & 255 & 0 & 0 & 0 & 255 & 255 \\
255 & 255 & 255 & 255 & 255 & 255 & 255 \\
255 & 255 & 255 & 255 & 255 & 255 & 255
\end{pmatrix}
$$

CONVOLUTIONAL NEURAL NETWORKS (CNNS)

Image representation on a computer:



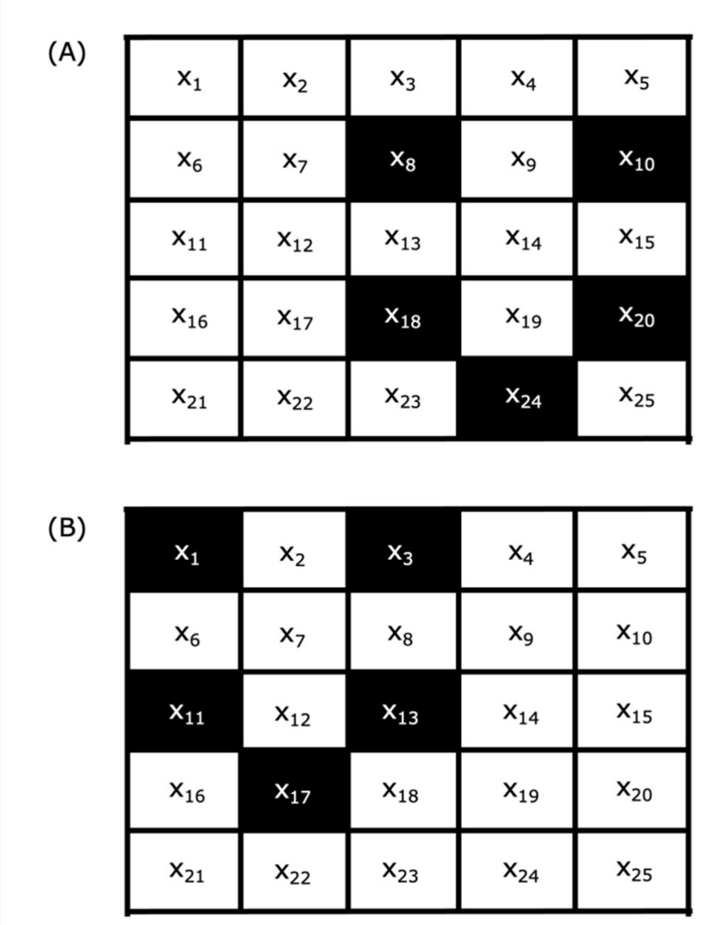
WHAT IS THE ISSUE WITH DENSE NEURAL NETWORKS WHEN WORKING WITH IMAGES?

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
1	25	3	7	0	0	0	70	0	0	0	150

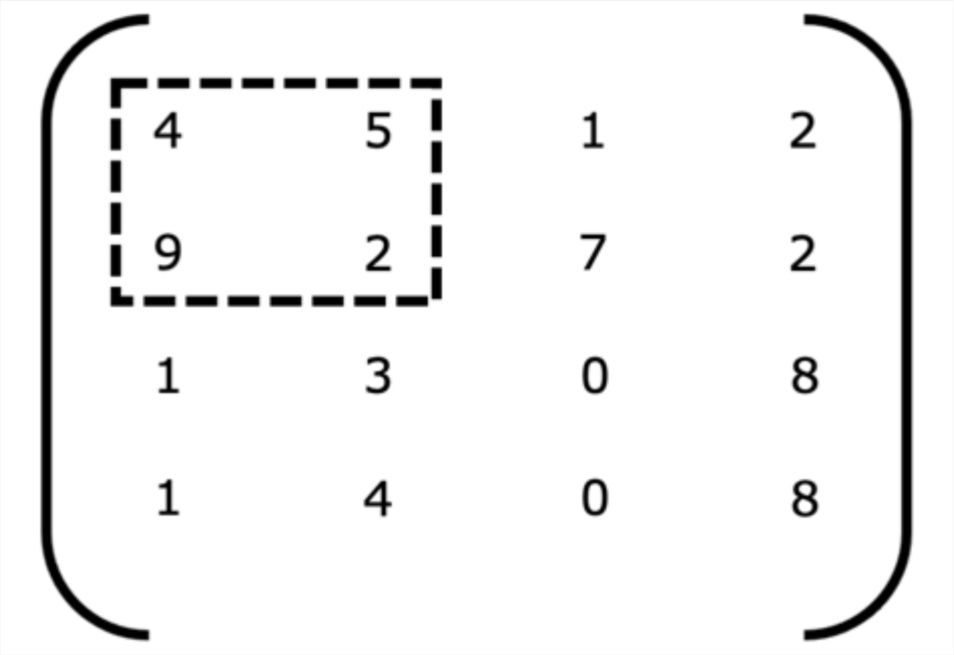
Red channel — Green channel — Blue channel

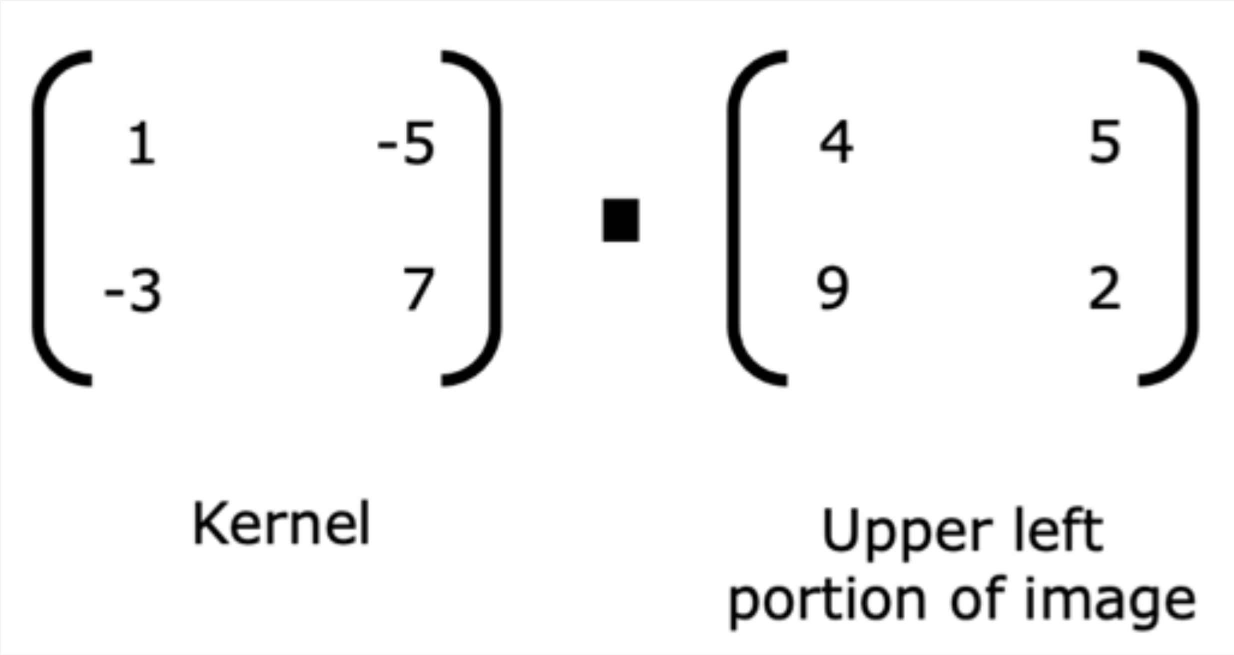
$$y = S(X) = \frac{1}{1 + e^{-(m_1 x_1 + \dots + m_{12} x_{12} + b)}}$$

TRANSLATIONAL INVARIANCE

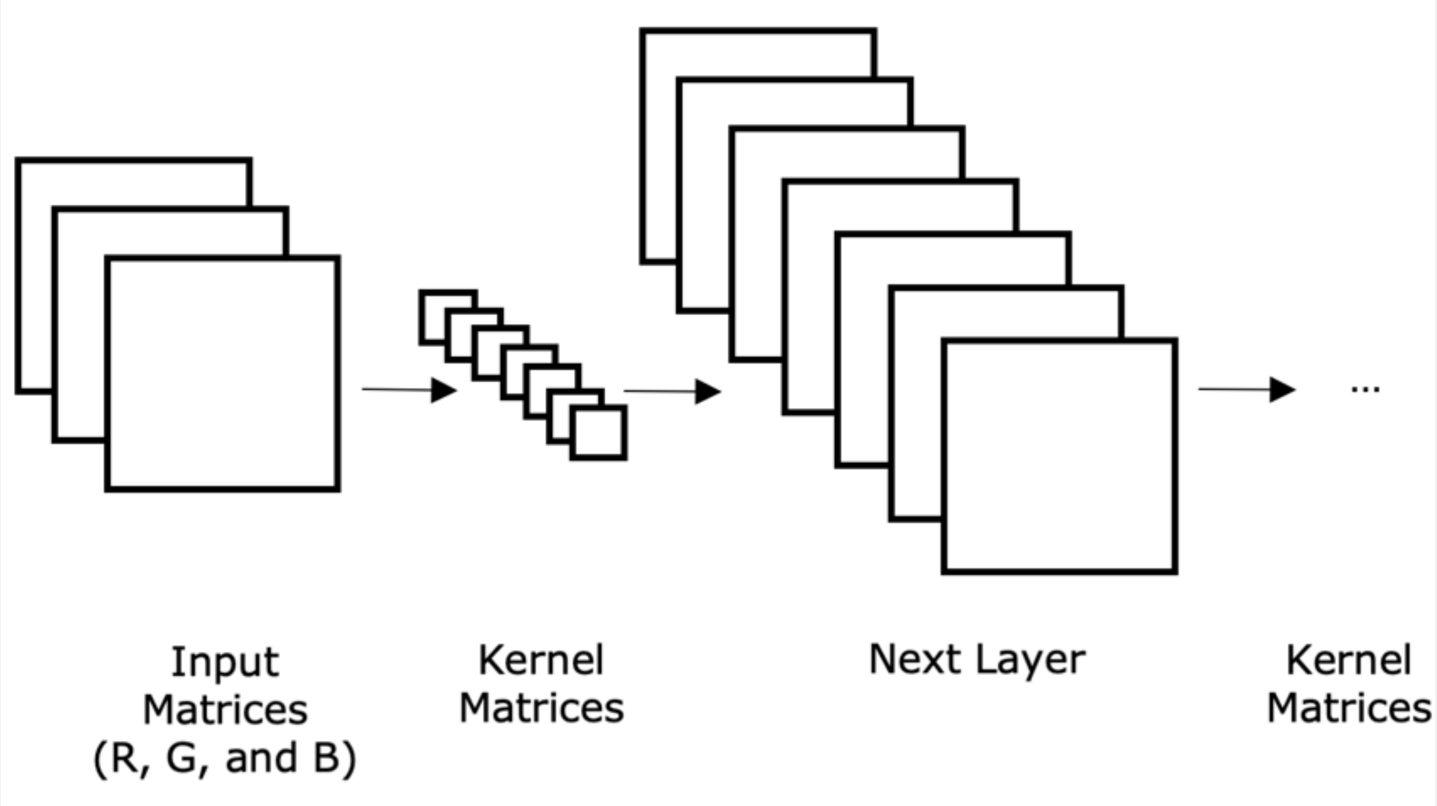


CONVOLUTIONAL NEURAL NETWORKS (CNNS)

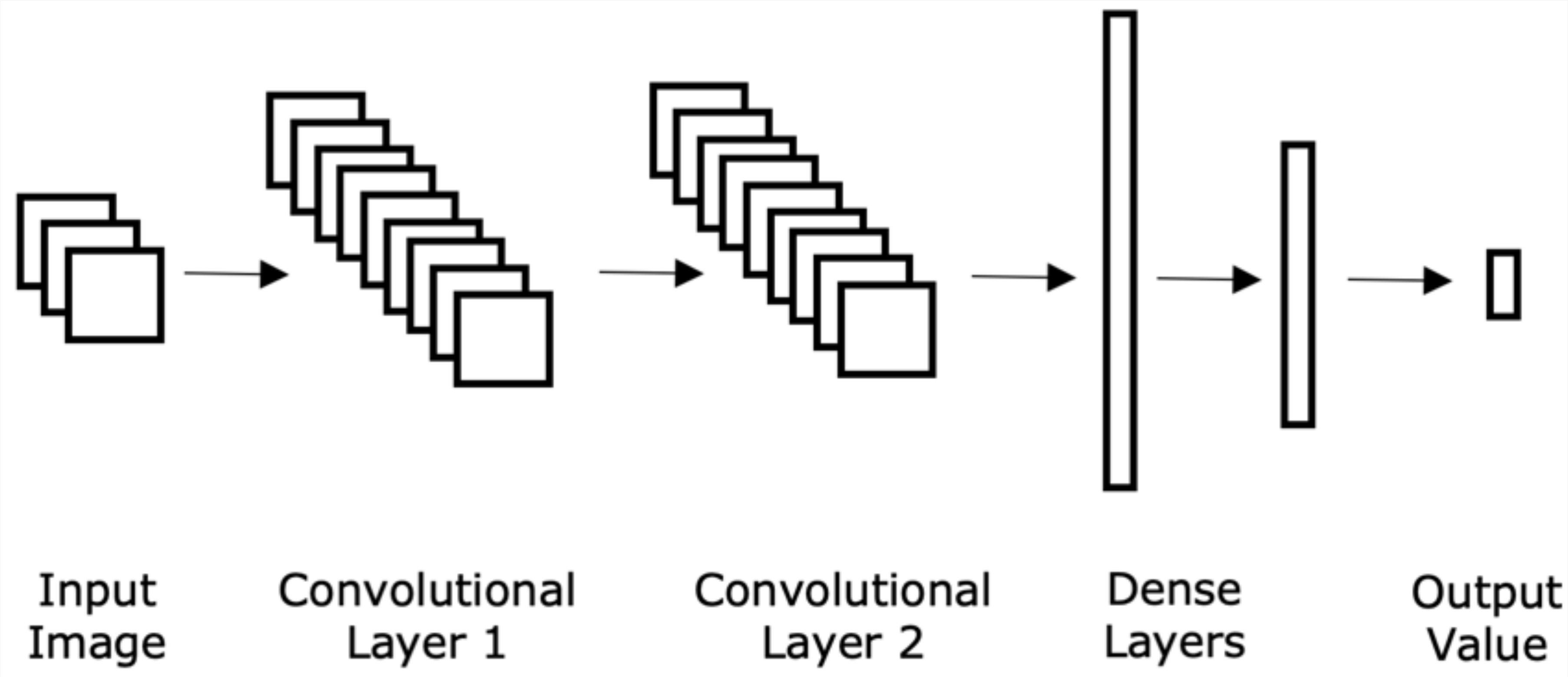




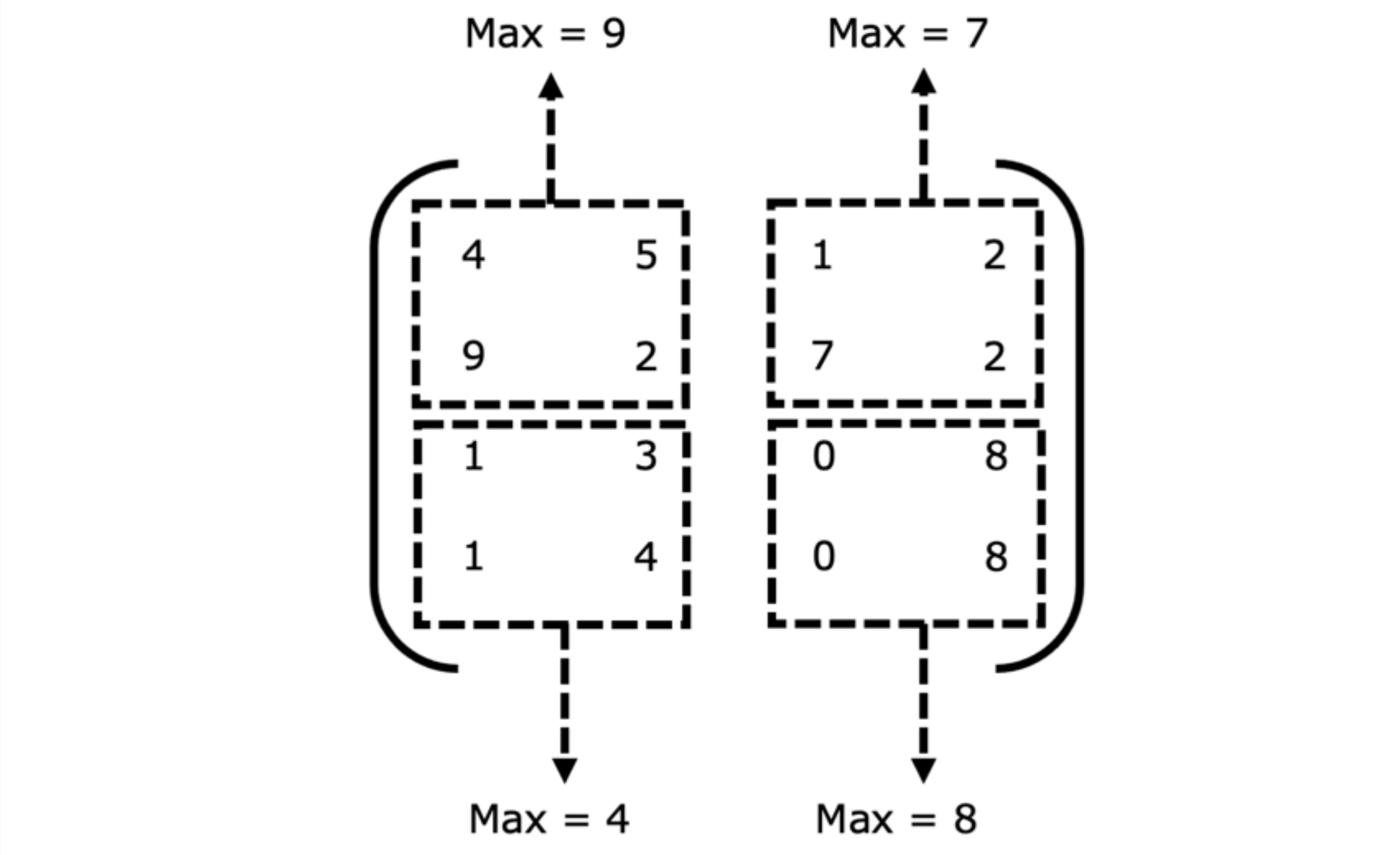
CONVOLUTIONAL NEURAL NETWORKS (CNNS)



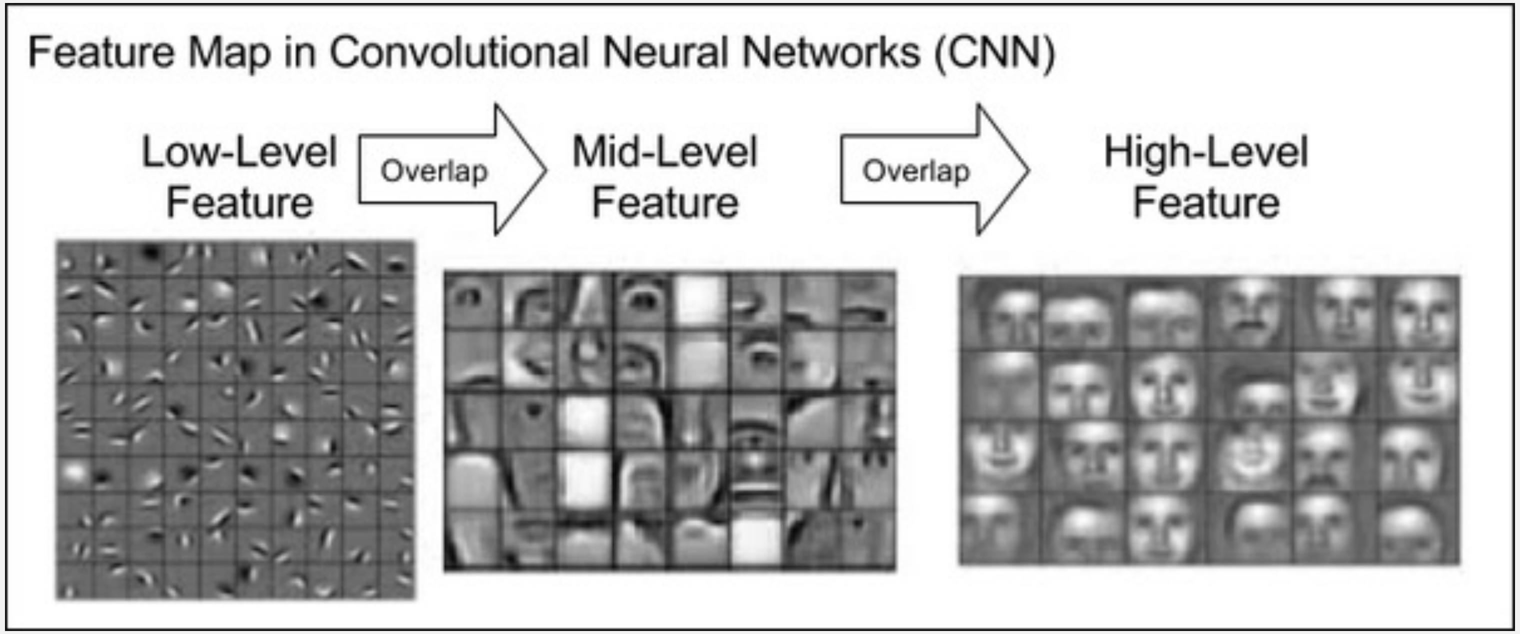
CONVOLUTIONAL NEURAL NETWORKS (CNNS)



CONVOLUTIONAL NEURAL NETWORKS (CNNS)

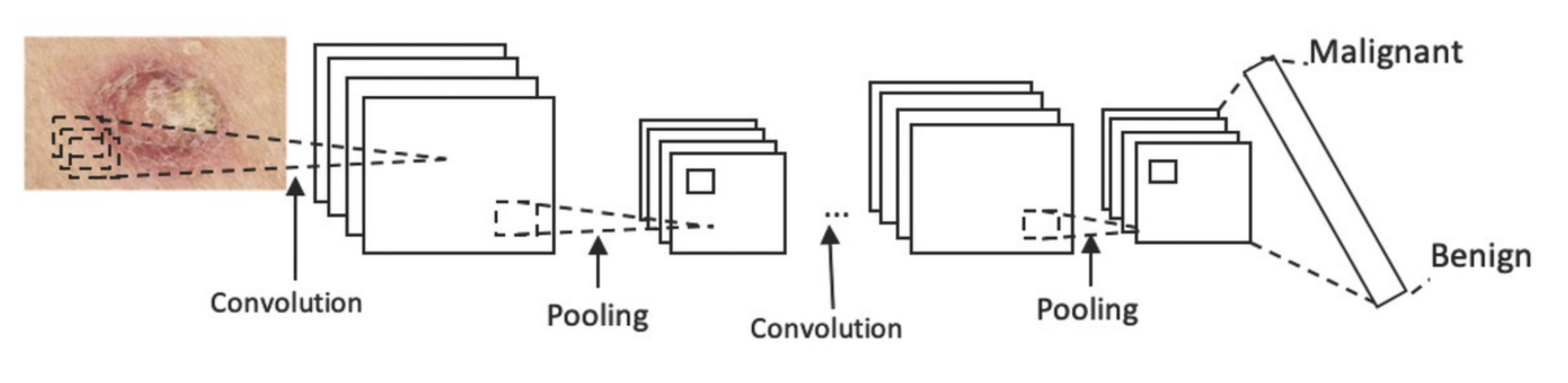


AUTOMATICALLY LEARNED FEATURE MAPS IN A CNN



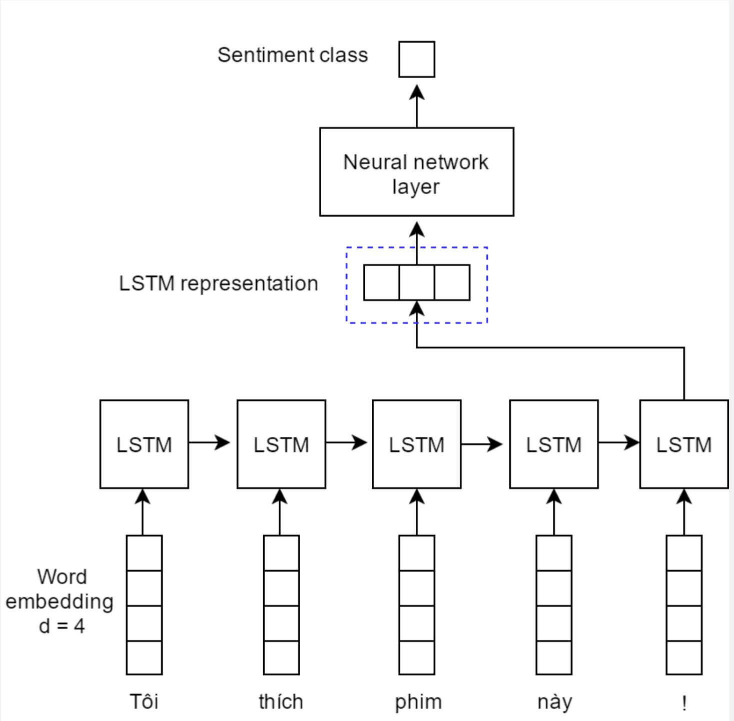
https://www.quora.com/How-does-a-convolutional-neural-network-recognize-an-occluded-face

EXAMPLE CNN APPLICATION: SKIN CANCER DETECTION



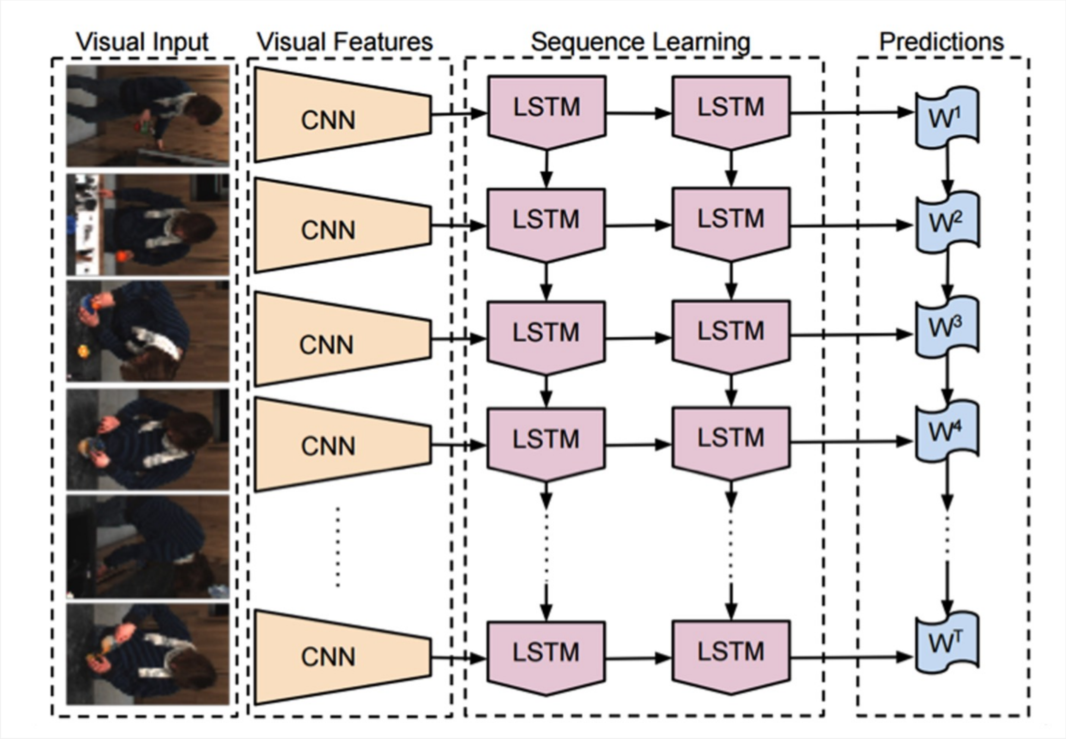
Zhang et al. "Optimization of the Convolutional Neural Networks for Automatic Detection of Skin Cancer." Open Medicine. 2020.

EXAMPLE RNN APPLICATION:



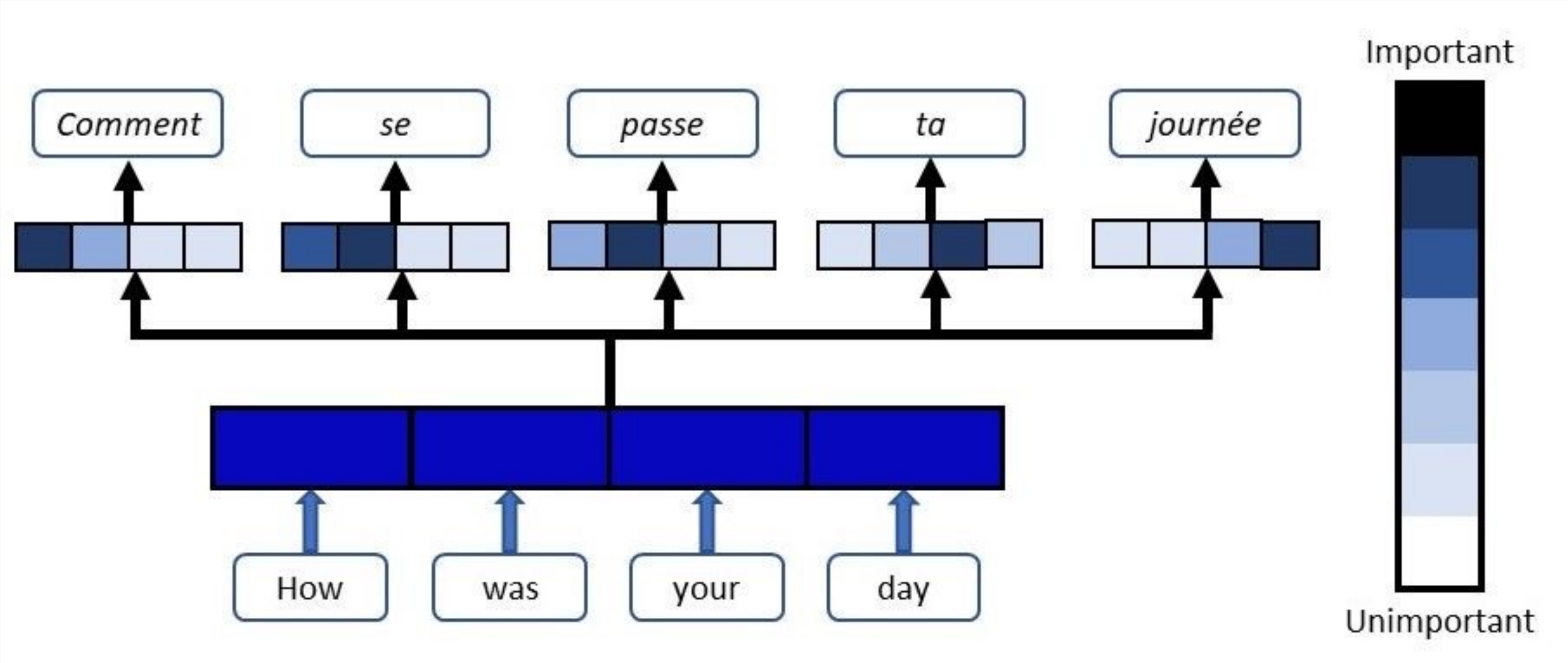
Vo et al. “Multi-channel LSTM-CNN model for Vietnamese sentiment analysis.” KSE. 2017.

EXAMPLE CNN+RNN APPLICATION:



Thung and Jiang. “ATorch Library for Action Recognition and Detection Using CNNs and LSTMs.” Stanford CS231N.

ATTENTION IS ALL YOU NEED



https://blog.floydhub.com/attention-mechanism/

TRANSFORMER MODELS

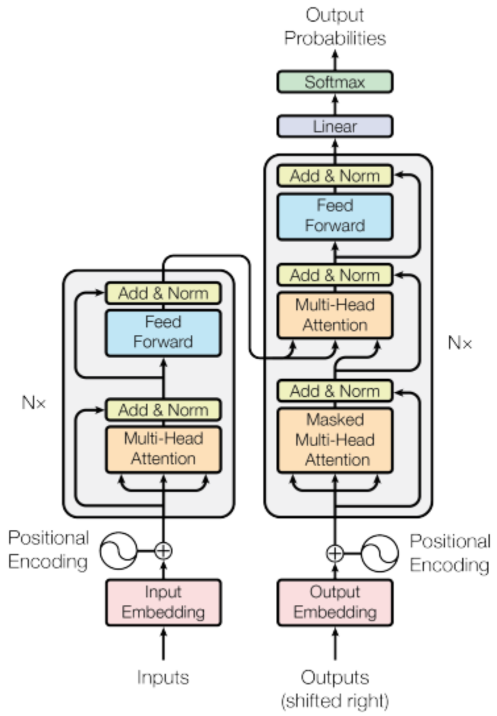
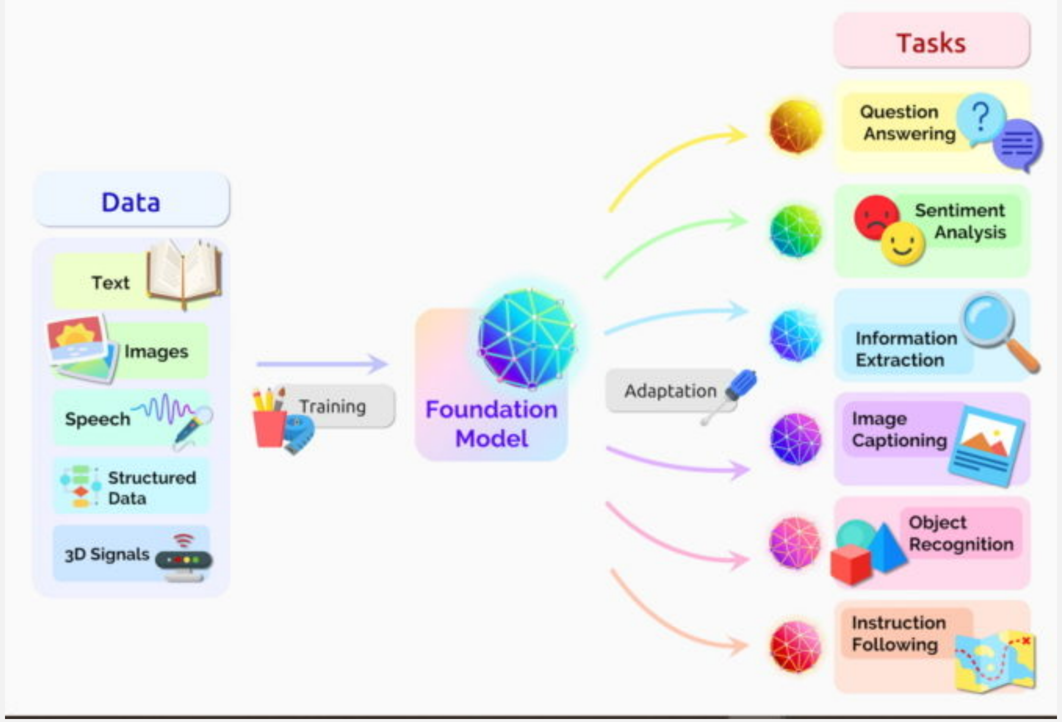


Figure 1: The Transformer - model architecture.

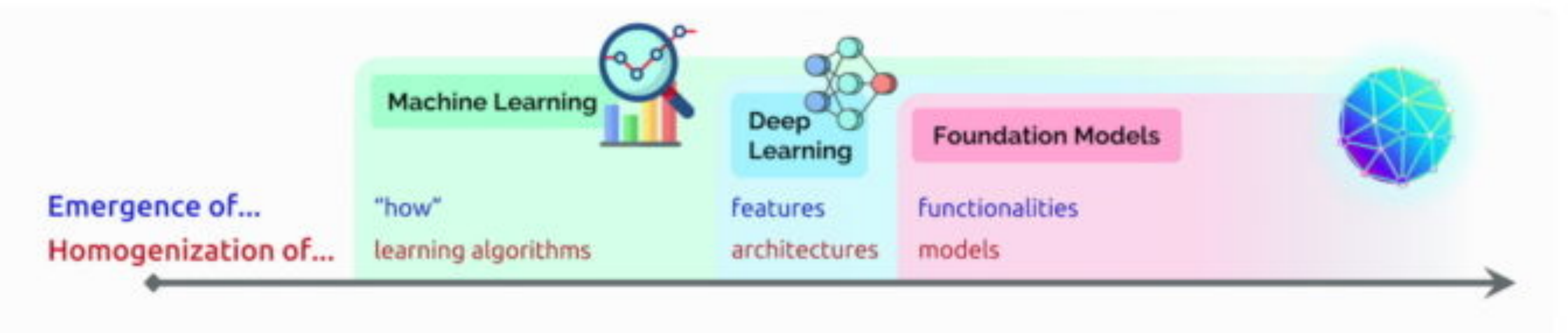
Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

USES OF TRANSFORMERS



https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/

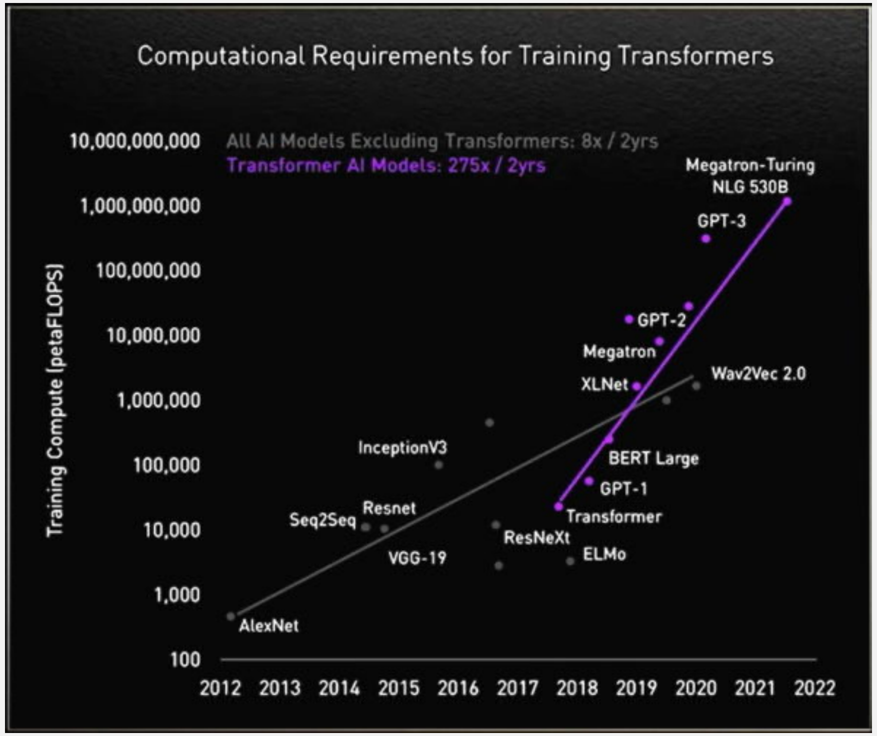
"FOUNDATION MODELS"



https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/

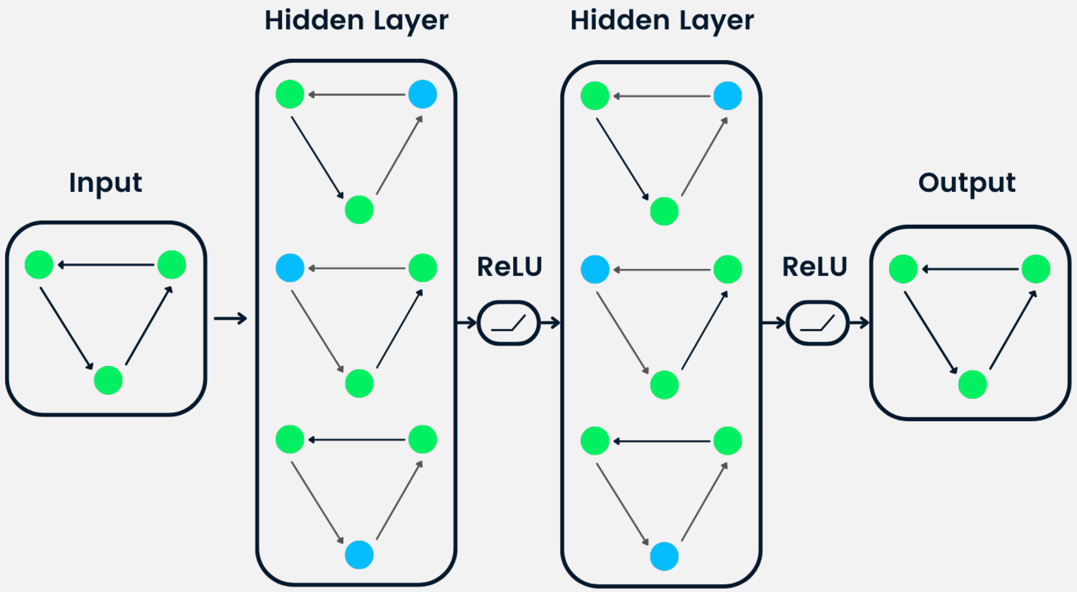
Foundation model:"model trained on a vast quantity of unlabeled data at scale (usually by self-supervised learning) resulting in a model that can be adapted to a wide range of downstream tasks" – Stanford HAI 2022

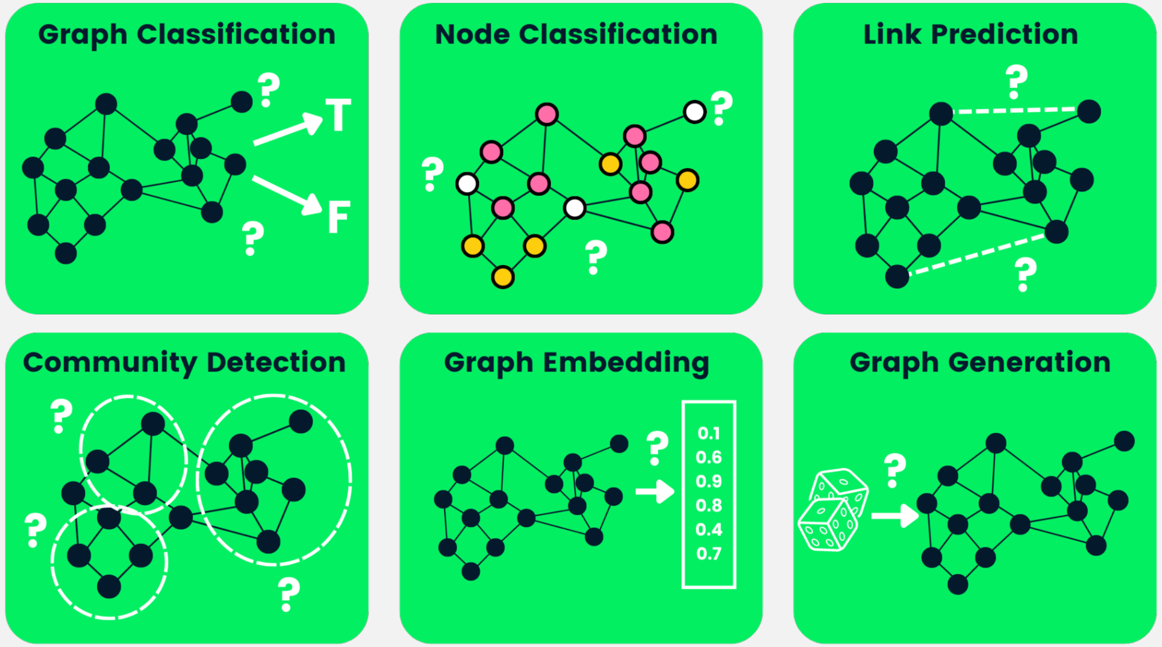
INCREASING COMPUTATIONAL REQUIREMENTS NEEDED TO TRAIN MODELS



https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/

GRAPH NEURAL NETWORKS (GNNS)



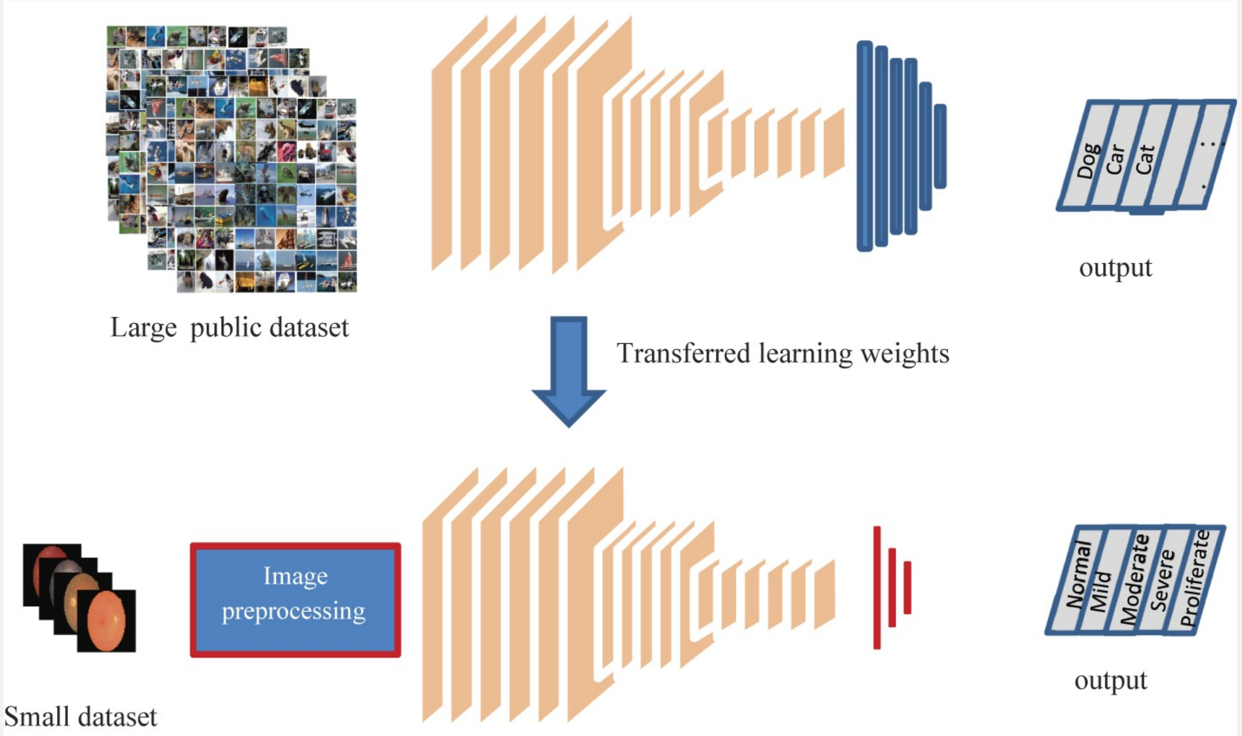


https://www.datacamp.com/tutorial/comprehensive-introduction-graph-neural-networks-gnns-tutorial

OUTLINE

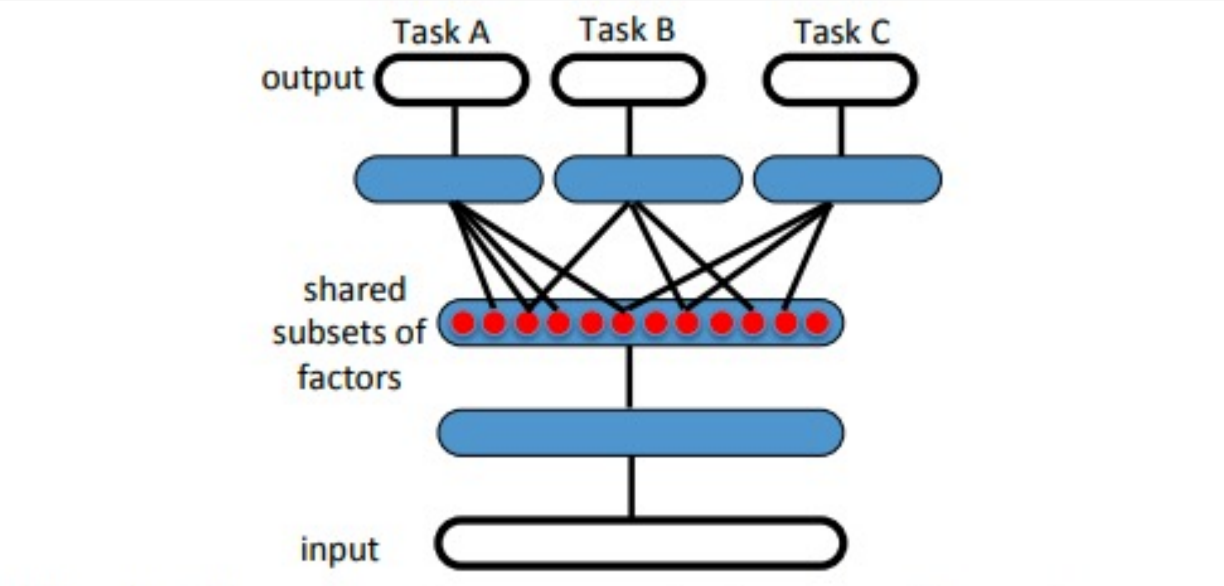
- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

TRANSFER LEARNING



https://www.emerald.com/insight/content/doi/10.1108/ACI-07-2021-0191/full/html

REPRESENTATION LEARNING

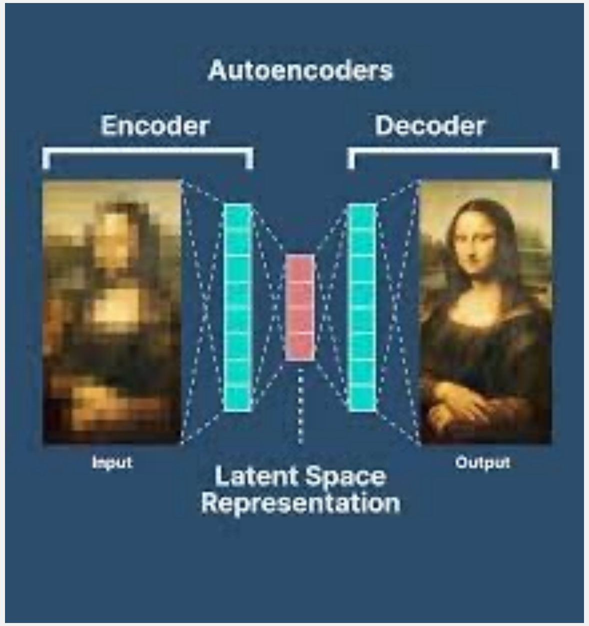


https://medium.com/@aganirbanghosh007/representation-learning-a-review-and-perspectives-ea923618d79c

OUTLINE

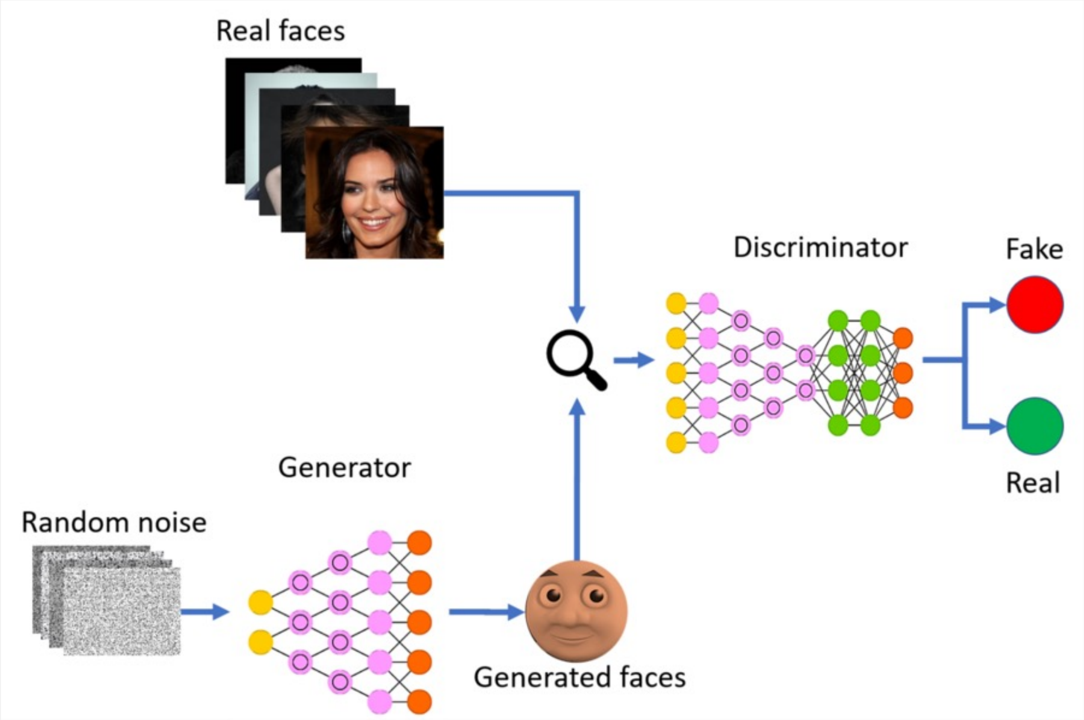
- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

AUTOENCODERS



https://www.v7labs.com/blog/autoencoders-guide

GENERATIVE ADVERSARIAL NETWORKS (GANS)



https://medium.com/sigmoid/a-brief-introduction-to-gans-and-how-to-code-them-2620ee465c30

OUTLINE

- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

TENSORFLOW / KERAS

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10)
])
```

CNN IN TENSORFLOW / KERAS

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu’))

model.add(layers.Dense(5, activation='relu'))))

model.add(layers.Dense(2), activation=‘sigmoid’)
```

TRANSFER LEARNING IN TENSORFLOW / KERAS

```
model = tf.keras.applications.MobileNetV2(input_shape=IMG_SHAPE,
                                          include_top=False,
                                          weights='imagenet’)

model.add(layers.Dense(5, activation='relu’)))
```

CODING DEMO

https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb

USEFUL EDUCATIONAL RESOURCES

- Deep Learning with Python, by François Chollet
- Free Deep Learning Courses – TensorFlow by Google
- DeepLearning.AI
- Deep Learning Basics: Introduction and Overview, by Lex Fridman
- https://colab.research.google.com/github/lexfridman/mit-deep-learning

PYTORCH

```
class Net(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = torch.flatten(x, 1) # flatten all dimensions except batch
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


net = Net()
```

OUTLINE

- Proposal Topics
- Neural Network Architectures
- Transfer Learning and Representation Learning
- Generative Networks
- Implementing DL in Python
- Discussion

DISCUSSION QUESTIONS

1. What are some interesting NN architectures you have encountered (or built)? What was the application?
2. What are various design decisions to be made when constructing a new neural network architecture? What factors affect these decisions?
3. What are some ethical considerations of transfer learning?
4. What are some inherent limitations of the left-to-right structure of out-of-the-box RNNs / LSTMs? How can these models be modified to support problems such as sequence-to-sequence classification (e.g., translation)?
5. How can CNNs and RNNs be combined? What are some examples of real-world applications where this would be useful?