IMPERIAL

Machine Learning for Neuroscience

ML4NS

Neuroscience-inspired Machine Learning and Applications in Neuroscience

Payam Barnaghi
Department of Brain Sciences &
School of Convergence Science in Human and Artificial Intelligence
Imperial College London
January 2025

1

1

Neuroscience and AI

IMPERIAL

- Neuroscience provides a rich source of ***inspiration*** for new types of algorithms and architectures.
- This is often independent of and complementary to the mathematical and logic-based methods and ideas that have largely dominated traditional approaches to AI.
- Neuroscience could help to provide a systems neuroscience-level view of the brain.
- It can help view the network, architecture, functions, and representations the brain utilises.
- The precise mechanisms by which the processes/interactions are physically realised in a biological substrate are often less relevant at the implementation level (of AI models).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

2

2

Transferrable ideas from neuroscience

IMPERIAL

- By focusing on the computational and algorithmic levels, we can obtain transferrable insights into general mechanisms of brain function while leaving room to accommodate the distinctive opportunities and challenges that arise when building intelligent machines *in silico*.
- For example, biological considerations informed the development of successful regularisation schemes that support generalisation beyond training data.
- One such scheme, in which only a subset of units participate in the processing of a given training example ("dropout"), was motivated by the stochasticity that is inherent in biological systems populated by neurons that fire with Poisson-like statistics (Hinton *et al.*, 2012).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

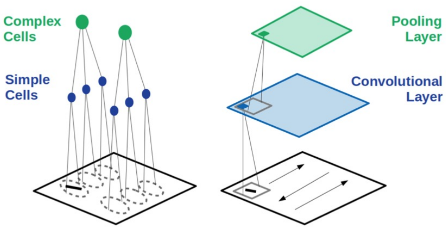
3

3

Deep learning and biological systems

IMPERIAL

- In both biological and artificial systems, successive non-linear computations transform raw visual input into an increasingly complex set of features, permitting object recognition that is invariant to transformations of pose, illumination, or scale.
- Hubel and Wiesel discovered that simple cells (left, blue) have preferred locations in the image (dashed ovals) wherein they respond most strongly to bars of particular orientation. Complex cells (green) receive input from many simple cells and thus have more spatially invariant responses.
-



Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017
Image source: Grace W. Lindsay, Gatsby Unit, UCL, https://arxiv.org/pdf/2001.07092.pdf

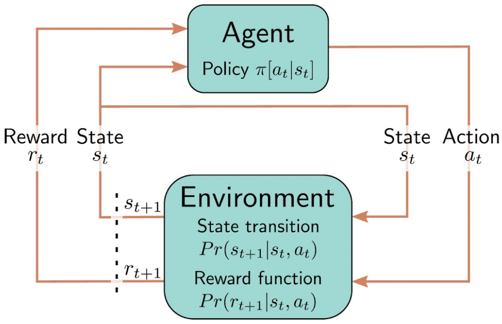
4

4

Reinforcement learning (RL)

IMPERIAL

Figure 19.6 Reinforcement learning loop. The agent takes an action a_t at time t based on the state s_t, according to the policy $\pi[a_t|s_t]$. This triggers the generation of a new state s_{t+1} (via the state transition function) and a reward r_{t+1} (via the reward function). Both are passed back to the agent, which then chooses a new action.



Source: Understanding Deep Learning, Simon J.D. Prince, MIT Press.

Reinforcement learning

IMPERIAL

- Reinforcement learning methods address the problem of how to maximise future reward by mapping states in the environment to actions and are among the most widely used tools in AI research (Sutton and Barto, 1998).
- RL methods were originally inspired by research into animal learning. In particular, developing temporal-difference (TD) methods, a critical component of many RL models, was inextricably intertwined with research into animal behaviour in conditioning experiments.
- Examples of neuroscience-informed RL models include TD methods and related techniques that have gone on to supply the core technology for recent advances in AI, ranging from robotic control (Hafner and Riedmiller, 2011) to expert play in backgammon (Tesauro, 1995) and Go (Silver *et al.*, 2016), and protein structures, AlphaFold (Jumper *et al.*, 2021).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

Reinforcement learning in AlphaFold

IMPERIAL

- Several components of AlphaFold's machine learning pipeline rely on supervised learning, such as predicting residue-residue distance and orientation from large datasets.
- However, reinforcement learning enhances predictions by providing feedback and implementing an iterative optimisation process.

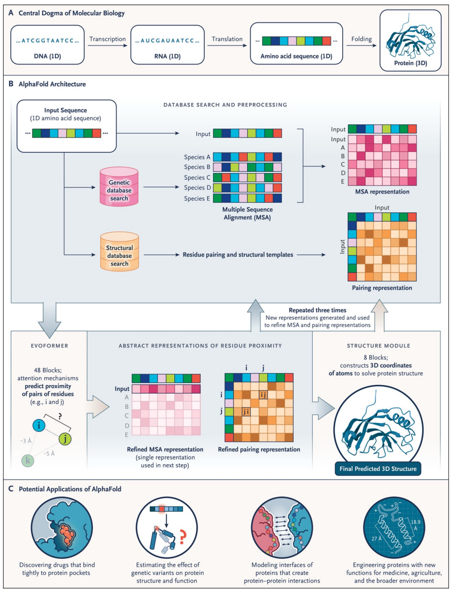


Image source: Altman RB. A Holy Grail - The Prediction of Protein Structure. N Engl J Med. 2023 Oct 12;389(15):1431-1434. doi: 10.1056/NEJMcibr2307735.

7

7

Attention-based models

IMPERIAL

- The brain does not learn by implementing a single, global optimisation principle within a uniform and undifferentiated neural network (Marblestone et al., 2016).
- Biological brains are modular, with distinct but interacting subsystems underpinning key functions such as memory, language, and cognitive control (Anderson et al., 2004; Shallice, 1988).
- One illustrative example is recent AI work on attention. Until recently, most CNN models worked directly on entire images or video frames, with equal priority given to all image pixels at the earliest processing stage. The primate visual system works differently.

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

8

8

CNNs and attention

IMPERIAL

- Rather than processing all input in parallel, visual attention shifts strategically among locations and objects, centring processing resources and representational coordinates on a series of regions in turn (Koch and Ullman, 1985; Moore and Zirnsak, 2017; Posner and Petersen, 1990).
- Detailed neurocomputational models have shown how this piecemeal approach benefits behaviour, by prioritising and isolating the information that is relevant at any given moment (Olshausen et al., 1993; Salinas and Abbott, 1997).
- One such network used this selective attentional mechanism to ignore irrelevant objects in a scene, allowing it to perform well in challenging object classification tasks in the presence of clutter (Mnih et al., 2014).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

9

9

Selective attention

IMPERIAL

Image source: TripAdvisor

10

10

Attention-based models and Transformers

IMPERIAL

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

11

11

Standard neural networks

IMPERIAL

- A standard neural network layer $f[x]$, takes a $D \times 1$ input x, and applies a linear transformation followed by an activation function $a[\bullet]$:

$$\mathbf{f}[\mathbf{x}] = \mathbf{a}[\boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{x}]$$

- Where, β contains the biases and Ω contains the weights.

12

12

A self-attention network*

IMPERIAL

- A self-attention block $sa[\bullet]$ takes N inputs x_n, each of dimension $D \times 1$, and returns N output vectors of the same size.
- In the context of NLP, each of the inputs x_n will represent a word or word fragment.
- First, a set of values are computed for each input:

$$\mathbf{v}_n = \boldsymbol{\beta}_v + \boldsymbol{\Omega}_v \mathbf{x}_n,$$

- where β_v and Ω_v represent biases and weights, respectively. Then the nth output $sa[xn]$ is a weighted sum of all the values v_n:

$$\mathbf{sa}[\mathbf{x}_n] = \sum_{m=1}^{N} a[\mathbf{x}_m, \mathbf{x}_n]\mathbf{v}_m.$$

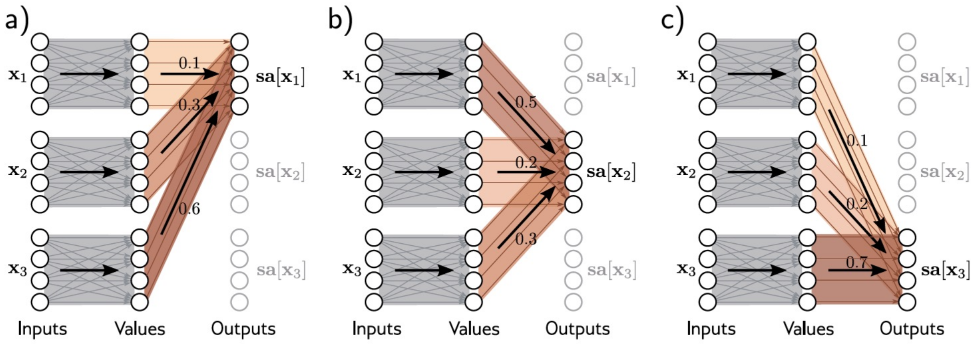
Understanding Deep Learning, Simon J.D. Prince, MIT Press.

13

13

Self-attention*

IMPERIAL



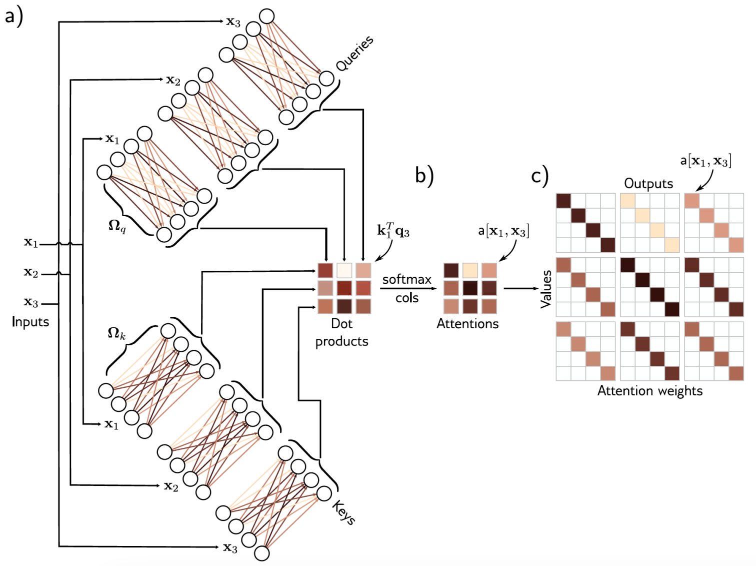
Understanding Deep Learning, Simon J.D. Prince, MIT Press.

14

14

Computing attention weights**

IMPERIAL



Computing attention weights**

IMPERIAL

– To compute the attention, we apply two more linear transformations to the inputs:

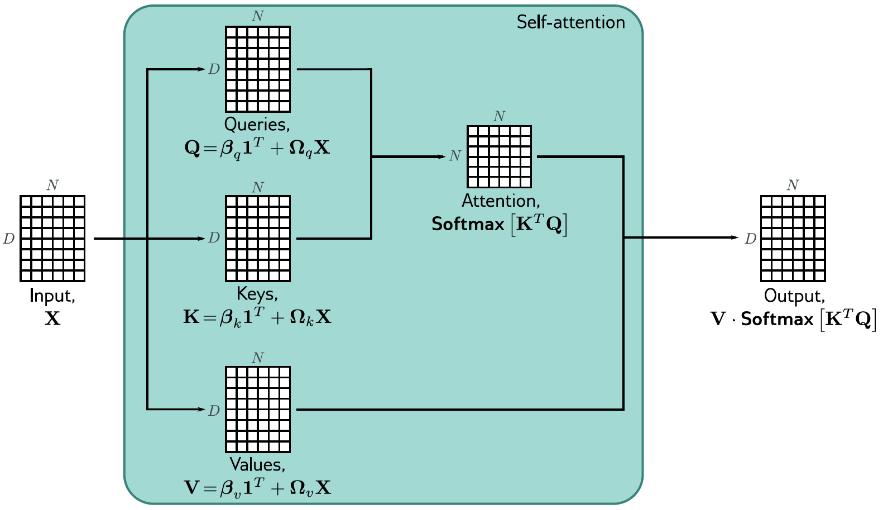
$$\begin{aligned}\mathbf{q}_n &= \boldsymbol{\beta}_q + \boldsymbol{\Omega}_q \mathbf{x}_n \\ \mathbf{k}_n &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{x}_n,\end{aligned}$$

– where q_n and k_k are referred to as queries and keys, respectively. Then we compute dot products between the queries and keys and pass the results through a softmax function.

$$\begin{aligned} a[\mathbf{x}_m, \mathbf{x}_n] &= \mathrm{softmax}_m \left[\mathbf{k}_\bullet^T \mathbf{q}_n\right] \\ &= \frac{\exp\left[\mathbf{k}_m^T \mathbf{q}_n\right]}{\sum_{m'=1}^{N} \exp\left[\mathbf{k}_{m'}^T \mathbf{q}_n\right]}, \end{aligned}$$

Self-attention in matrix form**

IMPERIAL



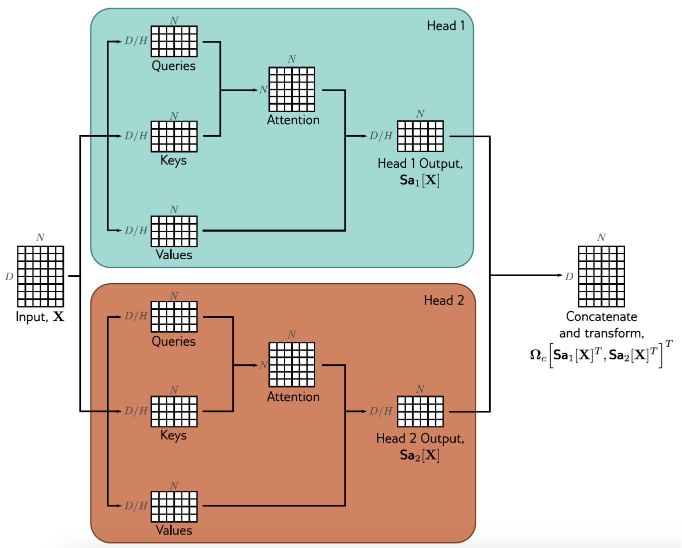
Understanding Deep Learning, Simon J.D. Prince, MIT Press.

17

17

Multi-head attention**

IMPERIAL



18

Understanding Deep Learning, Simon J.D. Prince, MIT Press.

18

Transformer layers*

IMPERIAL

- Self-attention is just one part of a larger transformer layer. This consists of a multi-head self-attention unit (which allows the word representations to interact with each other) followed by a fully connected network mlp[x•] (that operates separately on each word).
- Both units are residual networks (i.e., their output is added back to the original input).
- In addition, it is typical to add a LayerNorm operation after both the self-attention and fully connected networks.
- This is similar to BatchNorm but uses statistics across the tokens within a single input sequence to perform the normalization.

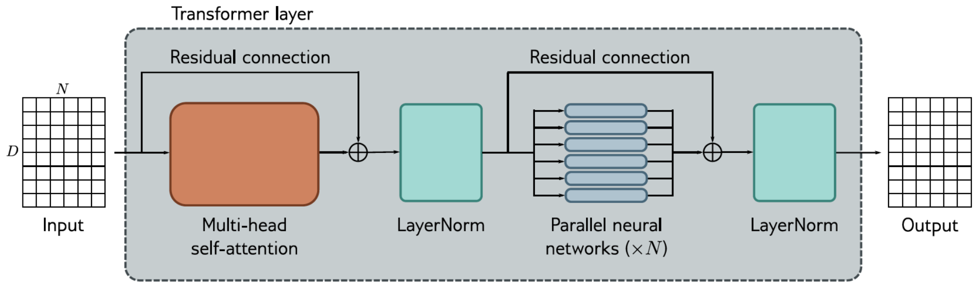
Understanding Deep Learning, Simon J.D. Prince, MIT Press.

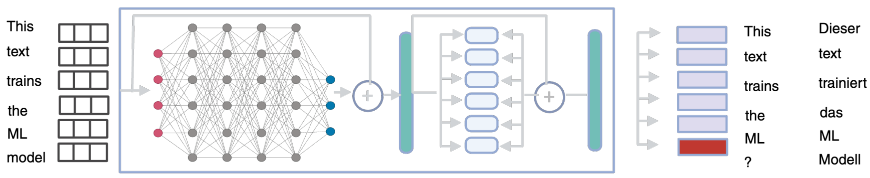
19

19

Transformer layers*

IMPERIAL





Understanding Deep Learning, Simon J.D. Prince, MIT Press.

20

20

Bidirectional Encoder Representations from Transformers (BERT)

IMPERIAL

- Encoder models like BERT exploit transfer learning.
- During pretraining, the parameters of the transformer architecture are learned using self-supervision from a large corpus of text.
- The goal here is for the model to learn general information about language statistics.
- In the fine-tuning stage, the resulting network is adapted to solve a particular task using a smaller body of supervised training data.

Understanding Deep Learning, Simon J.D. Prince, MIT Press.

21

21

Pre-training

IMPERIAL

- In the pre-training stage, the network is trained using self-supervision. This allows the use of enormous amounts of data without the need for manual labels.
- For BERT, the self supervision task consists of predicting missing words from sentences from a large internet corpus.
- During training, the maximum input length is 512 tokens, and the batch size is 256.
- The system is trained for a million steps, corresponding to roughly 50 epochs of the 3.3-billion word corpus.

22

22

Working memory

IMPERIAL

- Human intelligence is characterised by a remarkable ability to maintain and manipulate information within an active store, known as working memory. This memory is thought to be instantiated within the prefrontal cortex and interconnected areas (Goldman-Rakic, 1990).

- Classic cognitive theories suggest that this functionality depends on interactions between a central controller ("executive") and separate, domain-specific memory buffers (e.g., visuo-spatial sketchpad) (Baddeley, 2012).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

23

23

Continual learning

IMPERIAL

- Intelligent agents must be able to learn and remember many different tasks that are encountered over multiple timescales.
- Both biological and artificial agents must thus have a capacity for continual learning, that is, an ability to master new tasks without forgetting how to perform prior tasks (Thrun and Mitchell,1995). While animals appear relatively adept at continual learning, neural networks suffer from the problem of catastrophic forgetting (French, 1999; McClelland et al., 1995).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

24

24

Neuroimaging and continual learning

IMPERIAL

- In neuroscience, advanced neuroimaging techniques (e.g., two-photon imaging) now allow dynamic in vivo visualisation of the structure and function of dendritic spines during learning, at the spatial scale of single synapses (Nishiyama and Yasuda, 2015).
- This approach can be used to study neocortical plasticity during continual learning (Cichon and Gan, 2015; Hayashi-Takagi et al., 2015; Yang et al., 2009).

Source: Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-Inspired Artificial Intelligence. Neuron. 2017

AI in neuroscience

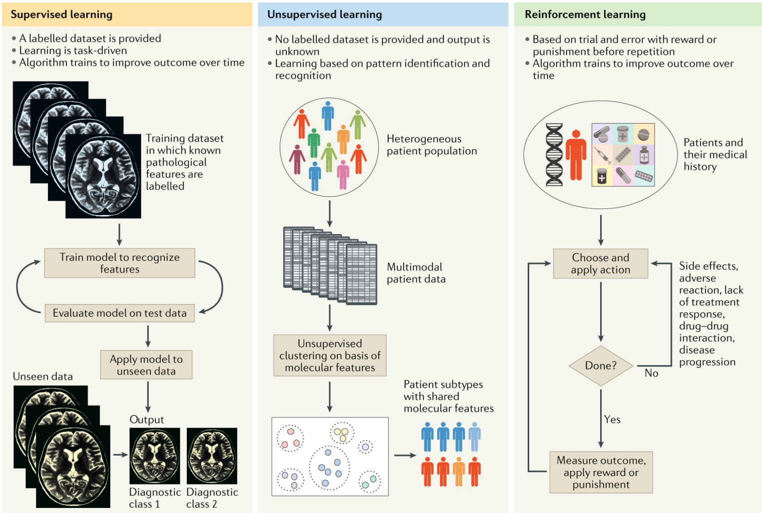
IMPERIAL

- Neuroimaging data analysis
- Behaviour and large population data analysis
- Computational models and simulations
- Graph and network analysis
- Fusion models and multimodal models for predictions/risk assessments

- An interesting read, How AI and neuroscience drive each other, https://www.nature.com/articles/d41586-019-02212-4

Applications of ML to diagnosis and treatment of neurodegenerative diseases

IMPERIAL



Source : Myszczynska, M.A., Ojamies, P.N., Lacoste, A.M.B. *et al.* Applications of machine learning to diagnosis and treatment of neurodegenerative diseases. *Nat Rev Neurol* 16, 440–456 (2020).

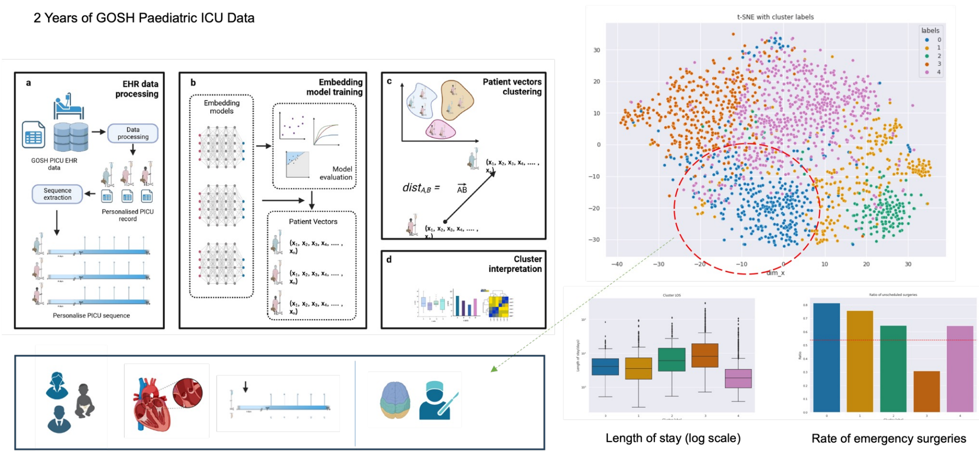
27

27

ML and Electronic Healthcare Records Analysis

IMPERIAL

2 Years of GOSH Paediatric ICU Data



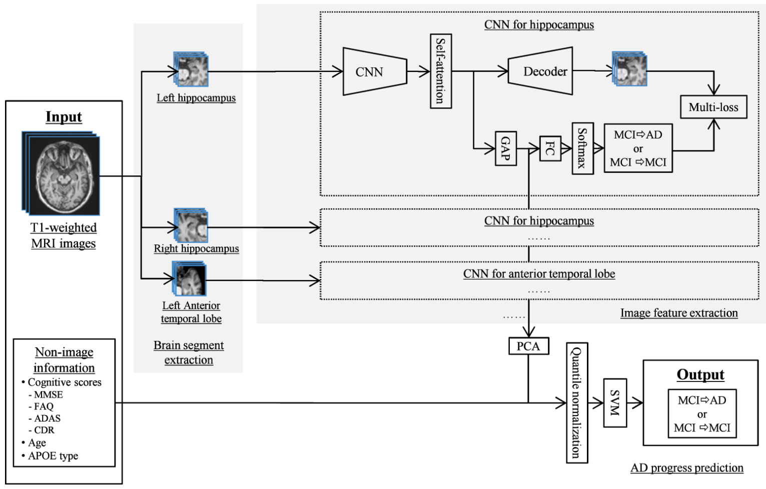
(J. Li *et al.*, 2025).

28

28

ML for predicting the progression of disease

IMPERIAL



Wang, C., Li, Y., Tsuboshita, Y. *et al.* A high-generalizability machine learning framework for predicting the progression of Alzheimer's disease using limited data.*npj Digit. Med.* 5, 43 (2022).

29

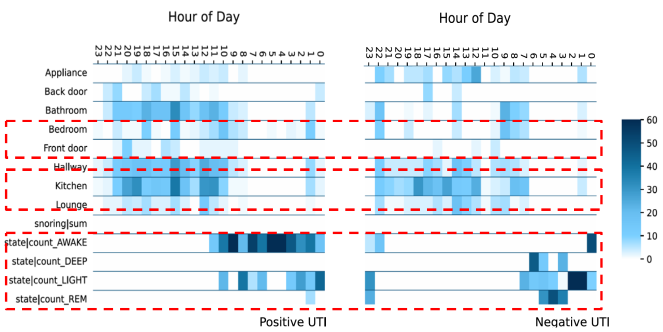
29

ML in precision health and care

IMPERIAL

UTIs are one of the top 5 reasons on unplanned hospital admissions in People Living with Dementia. UTI symptoms include increase in frequency of going to bathroom, delirium, increased temperature.

The model uses 20,738 person-days of in-home monitoring data which consists sleep, movement, and physiology data collected within the UK DRI CR&T study cohort (n=108) between 06/2021-07/2022.



(Alex Capstick *et al., npj Digital Medicine, 2024*).

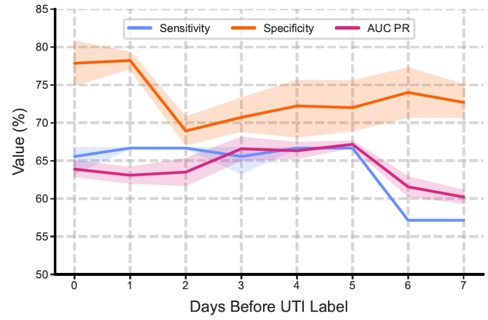
30

30

UTI risk analysis

IMPERIAL

		Sensitivity	Specificity	Precision
Date	Validation	86.6 (80.9 - 92.3)	94.5 (91.7 - 97.3)	87.3 (82.2 - 92.4)
Date	Test	69.0 (64.4 - 73.5)	94.1 (92.0 - 96.2)	81.9 (75.5 - 88.2)
Date-ID	Validation	98.3 (95.5 - 101.1)	90.0 (85.5 - 94.5)	81.7 (74.4 - 89.1)
Date-ID	Test	74.7 (67.9 - 81.5)	87.9 (85.0 - 90.9)	77.0 (71.9 - 82.1)



		Accuracy	No. of Participants	Positive : Negative	$\Pr(\hat{Y} = 1 \mid \text{Sex})$
Date	Female	52.6 (31.8 - 73.4)	16	1 : 1.7	35.5 (32.4 - 38.5)
	Male	86.5 (76.2 - 96.9)	25	1 : 5.5	32.3 (30.9 - 33.8)
Date-ID	Female	54.6 (26.8 - 82.4)	11	1 : 2.1	37.4 (33.6 - 41.1)
	Male	85.3 (72.9 - 97.7)	20	1 : 4.1	38.0 (36.4 - 39.7)

(Alex Capstick *et al., npj Digital Medicine, 2024*).

31

31

IMPERIAL

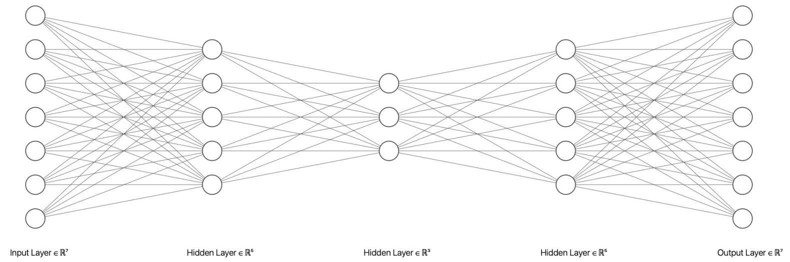
Some other practical/useful methods and techniques

32

Deep Autoencoders

IMPERIAL

- A typical autoencoder consists of an encoder and a decoder.
- The encoder projects the input to hidden representations and the decoder maps the hidden layer to the reconstruction layer.

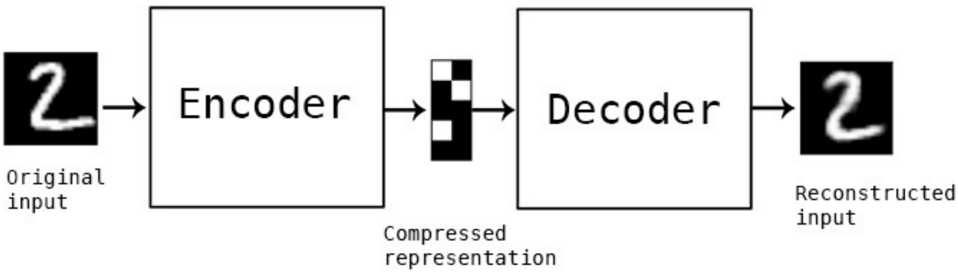


33

33

Autoencoder example

IMPERIAL



Source: https://blog.keras.io/building-autoencoders-in-keras.html

34

34

Autoencoders

IMPERIAL

- "Autoencoding" is a data representation algorithm in a lower dimension where the encoding and decoding functions:
  - data-specific
  - lossy,
  - *learned automatically from examples* rather than engineered by a human.
- Additionally, in almost all contexts where the term "autoencoder" is used, the encoder and decoder functions are implemented with neural networks.

Source: https://blog.keras.io/building-autoencoders-in-keras.html

35

35

Autoencoders

IMPERIAL

- Practical applications of autoencoders include **data denoising** and **dimensionality reduction**.
- With appropriate dimensionality and sparsity constraints, autoencoders can learn data projections that are more efficient than PCA or other basic techniques.

Source: https://blog.keras.io/building-autoencoders-in-keras.html

36

36

In other words

IMPERIAL

- An autoencoder is a neural network that is trained to attempt to copy its input to its output.
- Internally, it has a hidden layer h that describes a code (i.e. representation) used to represent the input.
- The network may be viewed as consisting of two parts: an encoder function $h = f(x)$ and a decoder that produces a reconstruction $r = g(h)$

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

37

37

Important note on autoencoders

IMPERIAL

- If an autoencoder succeeds in simply learning to set $g(f(x)) = x$ everywhere, then it is not especially useful.
- Instead, autoencoders are designed to be unable to learn to copy perfectly.
- Usually, autoencoders are restricted in ways that allow them to copy only approximately and to copy only input that resembles the training data. Because the model is forced to prioritise which aspects of the input should be copied, it often learns useful properties of the data.

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

38

38

Dropouts in autoencoders

IMPERIAL

– You can add dropout to reduce overfitting.

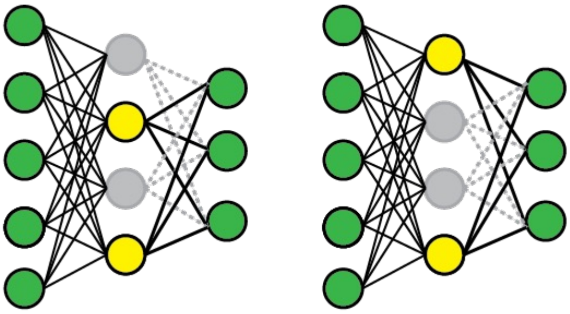


Image source: Matt Krause via https://stats.stackexchange.com/questions/201569/what-is-the-difference-between-dropout-and-drop-connect

39

39

Undercomplete representation

IMPERIAL

– An autoencoder whose code dimension is less than the input dimension is called undercomplete.

– Learning an under-complete representation forces the autoencoder to capture the most salient features of the training data

40

40

Learning process in autoencoders

IMPERIAL

- The learning process is described simply as minimising a loss function:

$$L(x, g(f(x)))$$

- where where L is a loss function penalising $g(f(x))$ for being dissimilar from x such as the mean squared error.

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

41

41

Linear decoders

IMPERIAL

- When the decoder is linear and L is the mean squared error, an undercomplete autoencoder learns to span the same subspace as PCA.
- In this case, an autoencoder trained to perform the copying task has learned the principal subspace of the training data as a side effect.

- Autoencoders with nonlinear encoder functions f and nonlinear decoder functions g can thus learn a more powerful nonlinear generalisation of PCA.

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

42

42

Sparse autoencoders

IMPERIAL

- A sparse autoencoder is simply an autoencoder whose training criterion involves a sparsity penalty $\Omega(h)$ on the code layer h, in addition to the reconstruction error:

$$L(x, g(f(x)) + \Omega(h)$$

- Sparse autoencoders are typically used to learn features for another task, such as classification. An autoencoder that has been regularised to be sparse must respond to unique statistical features of the dataset it has been trained on, rather than simply acting as an identity function. In this way, training to perform the copying task with a sparsity penalty can yield a model that has learned useful features as a by product.

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

43

43

Sparse autoencoders

IMPERIAL

- We can think of the penalty $\Omega(h)$ simply as a regularizer term added to a feedforward network whose primary task is to copy the input to the output (unsupervised learning objective) and possibly also perform some supervised task (with a supervised learning objective) that depends on these sparse features.

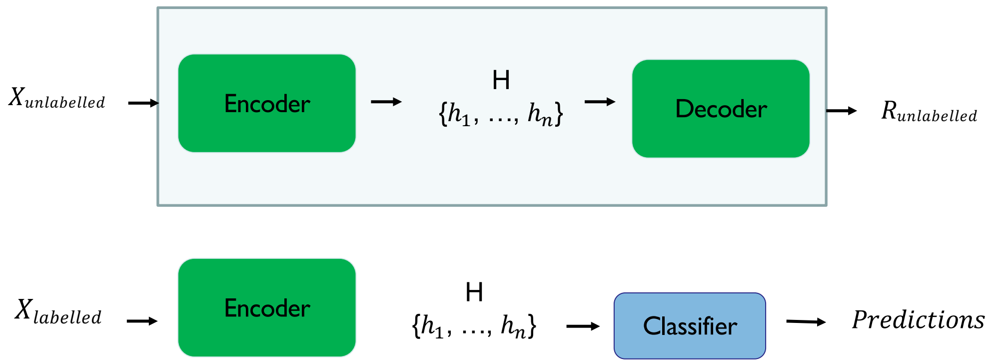
Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

44

44

A semi-supervised learning scenario

IMPERIAL



45

45

Encoding categorical features

IMPERIAL

- Often, features are not given as continuous values but categorical.
- For example, a person could have features ["male", "female"], [”low", ”mild” ”high", ”severe"]..
- Such features can be efficiently coded as integers; for instance, ["male",”has low symptoms"] could be expressed as [0, 1] while ["female", ”has severe"] would be [1, 4].
- Such integer representation often cannot be used directly with machine learning models.

46

46

Encoding categorical features

IMPERIAL

- Such integer representation cannot be used directly with *machine learning models that* expect continuous input and would interpret the categories as being ordered, which is often not desired (i.e. the set of browsers was ordered arbitrarily).
- One possibility to convert categorical features to features that can be used with *scikit-learn* estimators is to use a *one-of-K* or *one-hot* encoding, which is implemented in *OneHotEncoder*.
- This estimator transforms each categorical feature with *m* possible values into *m* binary features, with only one active.

47

47

Encoding categorical features

IMPERIAL

- By default, the number of values each feature can take is inferred automatically from the dataset.
- It is possible to specify this explicitly using the parameter *n_values*.
- For example, if there are two genders, three possible continents and four web browsers in our dataset. Then, we fit the estimator and transform a data point.
- In the result, the first two numbers encode the gender, the next set of three numbers is the continent and the last four are the web browser.

48

48

OneHotEncoder

IMPERIAL

- The input to this transformer should be a matrix of integers, denoting the values taken on by categorical (discrete) features.
- The output will be a sparse matrix where each column corresponds to one possible value of one feature.
- It is assumed that input features take on values in the range [0, n_values).
- This encoding is needed for feeding categorical data to many scikit-learn estimators, notably linear models and SVMs with the standard kernels.

49

49

OneHotEncoder- example

IMPERIAL

50

50

Imputation of missing values

IMPERIAL

- For various reasons, many real-world datasets contain missing values, often encoded as blanks, NA or other placeholders.
- Such datasets, however, are incompatible with scikit-learn estimators, which assume that all values in an array are numerical and that all have and hold meaning.
- A basic strategy to use incomplete datasets is to discard entire rows and/or columns containing missing values.
- However, this comes at the price of losing data which may be valuable (even though incomplete).
- A better strategy is to impute the missing values, i.e., to infer them from the known part of the data.

51

51

The *Imputer* class (in Python)

IMPERIAL

- The *Imputer* class provides basic strategies for imputing missing values, either using the mean, the median or the most frequent value of the row or column in which the missing values are located.
- This class also allows for different missing values encodings.

- **strategy**: string, optional (default="mean")
- The imputation strategy.
  - If "*mean*", then replace missing values using the mean along the axis.
  - If "*median*", then replace missing values using the median along the axis.
  - If "*most_frequent*", then replace missing using the most frequent value along the axis.

52

52

Data imputation

IMPERIAL

- There are obviously other more sophisticated imputation techniques.
- However, the most important consideration is knowing the data, application context, and the techniques and solutions that would be more relevant to the given data/context and those that won't add more bias to the data.

53

53

Feature selection

IMPERIAL

- The classes in the *sklearn.feature_selection* module can be used for feature selection/dimensionality reduction on sample sets, either to improve estimators' accuracy scores or to boost their performance on very high-dimensional datasets.

54

54

Removing features with low variance

IMPERIAL

- *VarianceThreshold* is a simple baseline approach to feature selection.
- It removes all features whose variance doesn't meet some threshold. By default, it removes all zero-variance features, i.e. features that have the same value in all samples.
- There are several other methods/techniques and models for feature selection. We will/have studied some of those in this series.

55

55

IMPERIAL

Some of the topics that we didn't cover in this module

(some of these will be covered in the optional series in March/April)

56

56

Sampling and Monte Carlo methods

IMPERIAL

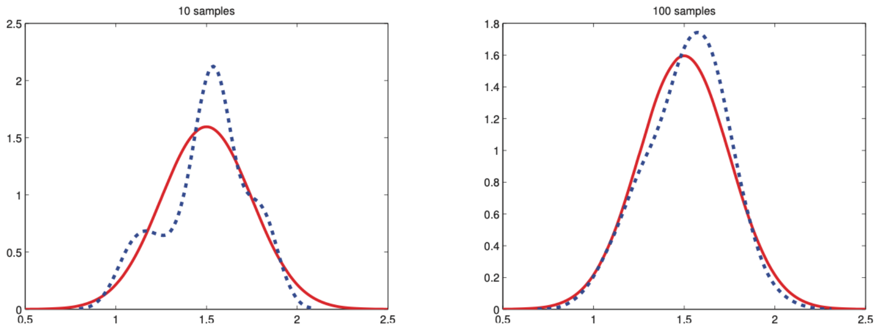


Image source: Kevin Murphy, Machine Learning: A probabilistic perspective

Example work on healthcare applications, Pourshahrokhi *et al.*, https://arxiv.org/pdf/2103.02349.pdf

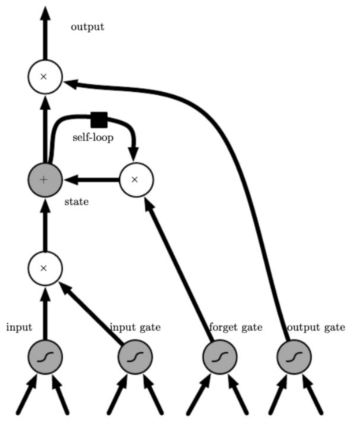
57

57

Recurrent models

IMPERIAL

– A view of Long Short Term Memory (LSTM) model



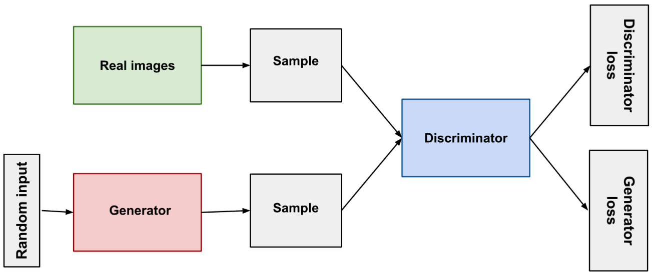
Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

58

58

Generative Adversarial Networks*

IMPERIAL



- Both the generator and the discriminator are neural networks.
- The generator output is connected directly to the discriminator input. Through backpropagation, the discriminator's classification provides a signal that the generator uses to update its weights.

Source: Ian Goodfellow et al., Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org

59

59

Diffusion models*

IMPERIAL

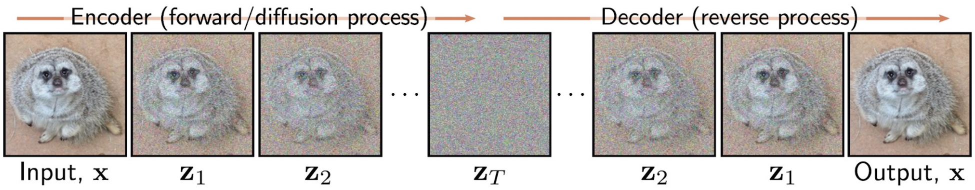


Figure 18.1 Diffusion models. The encoder (forward, or diffusion process) maps the input $\mathbf{x}$ through a series of latent variables $\mathbf{z}_1 \ldots \mathbf{z}_T$. This process is pre-specified and gradually mixes the data with noise until only noise remains. The decoder (reverse process) is learned and passes the data back through the latent variables, removing noise at each stage. After training, new examples are generated by sampling noise vectors $\mathbf{z}_T$ and passing them through the decoder.

Source: Understanding Deep Learning, Simon J.D. Prince, MIT Press.

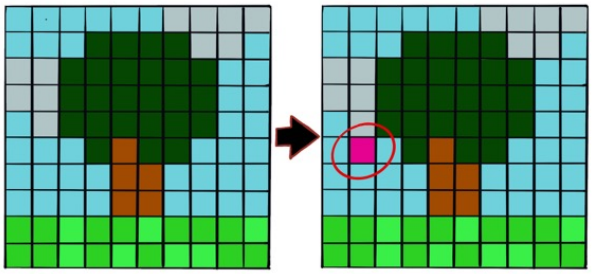
60

60

Adversarial Examples

IMPERIAL

– Can we change one pixel in an image to deceive a machine learning model?



Original paper: *Su, Jiawei, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks. "IEEE Transactions on Evolutionary Computation (2019)*

Image source: https://christophm.github.io/interpretable-ml-book/adversarial.html

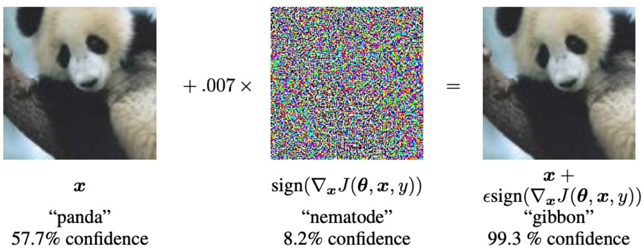
61

61

A linear adversarial example*

IMPERIAL

– A demonstration of fast adversarial example generation applied to GoogLeNet(Szegedy et al., 2014) on ImageNet.



Original paper: Explaining and Harnessing Adversarial Examples, Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, https://arxiv.org/abs/1412.6572

62

62

Training Large Language Models (LLMs)

IMPERIAL

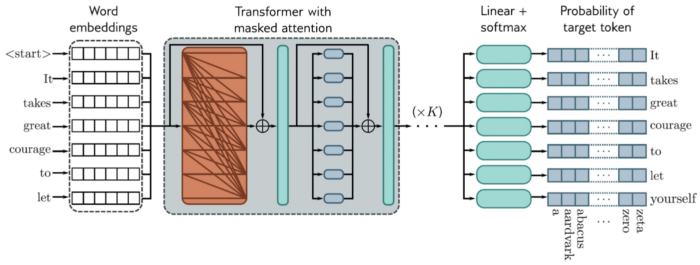


Figure 12.12 Training GPT3-type decoder network. The tokens are mapped to word embeddings with a special <start> token at the beginning of the sequence. The embeddings are passed through a series of transformers that use masked self-attention. Here, each position in the sentence can only attend to its own embedding and the embeddings of tokens earlier in the sequence (orange connections). The goal at each position is to maximize the probability of the following ground truth token in the sequence. In other words, at position one, we want to maximize the probability of the token It; at position two, we want to maximize the probability of the token takes; and so on. Masked self-attention ensures the system cannot cheat by looking at subsequent inputs. The autoregressive task has the advantage of making efficient use of the data since every word contributes a term to the loss function. However, it only exploits the left context of each word.

Source: Understanding Deep Learning, Simon J.D. Prince, MIT Press.

63

63

If you have any questions

IMPERIAL

– Please feel free to arrange a meeting or email (p.barnaghi@imperial.ac.uk).
– My office: 928, Sir Michael Uren Research Hub, White City Campus.

64

64