UNIVERSITY OF
OXFORD

Mathematical
Institute

Why deep learning: application and approximation.

Theories of Deep Learning: C6.5,
Lecture / Video 2

Prof. Jared Tanner
Mathematical Institute
University of Oxford

Oxford
Mathematics

Recent breakthroughs in deep learning applications:

Much of why we are interested in deep learning is its remarkable efficacy

Deep learning is now widespread in applications, showing remarkable abilities to perform complex tasks.

- Computer vision; image classification, Imagenet challenges.
- Complex strategy games such as Go.
- AI for art with style transfer, sound generation from videos, and text generation.
- Deep learning is now increasingly used in scientific applications: gravitational lensing and weather prediction (DGMR),
- AlphaFold for protein structure interaction prediction.
- And many many more applications: e.g. medical diagnostics....

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

Image classification, localization, and detection:

http://image-net.org

ImageNet was first presented in 2009 to help benchmark image classification algorithms in the ILSVRC.

2010-14: Image classification; 1.2 million training labeled images

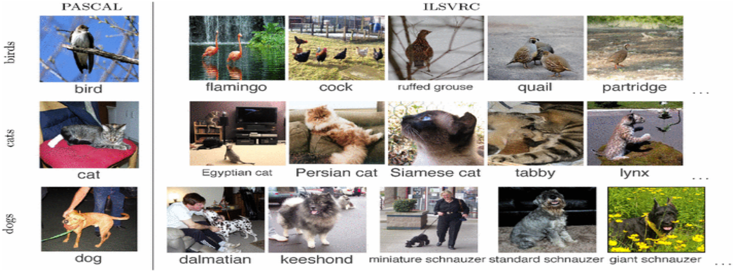
2011-14: Single object localisation; 524,000 training labeled bbox

2013-14: All object classification per scene; 456,000 training set

https://link.springer.com/article/10.1007/s11263-015-0816-y

ImageNet Large Scale Visual Recognition Challenge

Image classification, localization, and detection: complex set of similar data

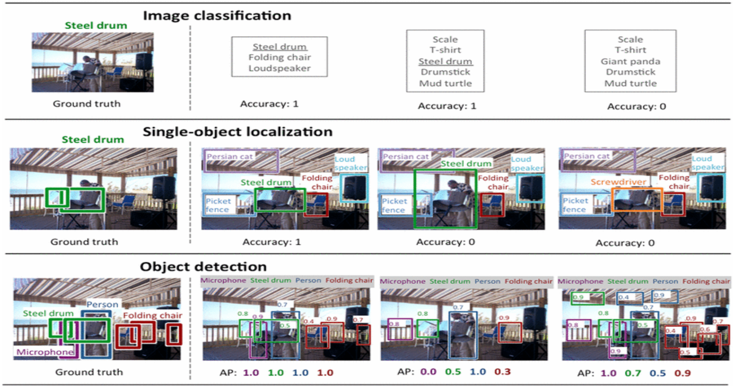


"The ILSVRC dataset contains many more fine-grained classes compared to the standard PASCAL VOC benchmark; for example, instead of the PASCAL "dog" category there are 120 different breeds of dogs in ILSVRC2012-2014 classification and single-object localization tasks."

https://link.springer.com/article/10.1007/s11263-015-0816-y

ImageNet Large Scale Visual Recognition Challenge

Image classification, localization, and detection

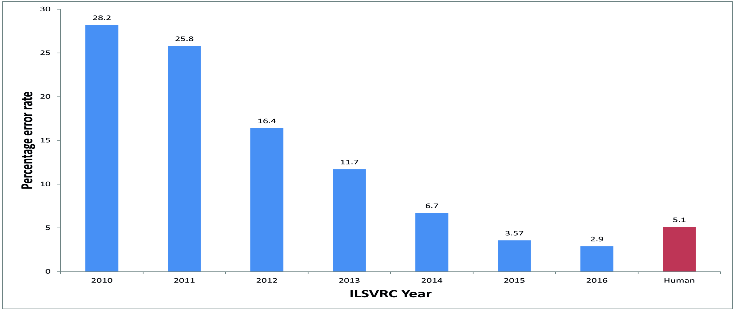


2013-14: All object classification per scene; 456,000 training set

https://arxiv.org/pdf/1409.0575.pdf

ImageNet Large Scale Visual Recognition Challenge

Image classification, localization, and detection: "super-human performance."

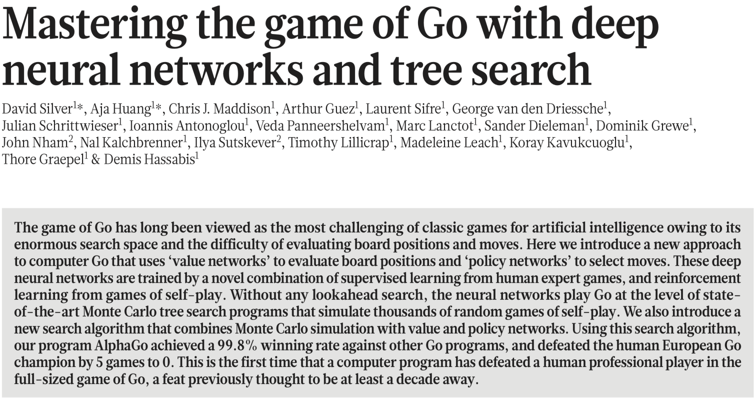


https://arxiv.org/pdf/1409.0575.pdf

2012 ILSVRC classification won using 7 layer CNN by Krizhevsky, Sutskever, and Hinton; users in widespread use of ConvNets. This success marked the start of DNNs widespread use.

Deep learning for games

Improved with reinforcement learning.



Mastering the game of Go with deep neural networks and tree search

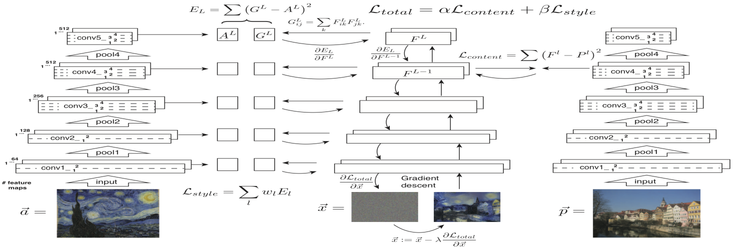
David Silver[1]*, Aja Huang[1]*, Chris J. Maddison[1], Arthur Guez[1], Laurent Sifre[1], George van den Driessche[1], Julian Schrittwieser[1], Ioannis Antonoglou[1], Veda Panneershelvam[1], Marc Lanctot[1], Sander Dieleman[1], Dominik Grewe[1], John Nham[2], Nal Kalchbrenner[1], Ilya Sutskever[2], Timothy Lillicrap[1], Madeleine Leach[1], Koray Kavukcuoglu[1], Thore Graepel[1] & Demis Hassabis[1]

The game of Go has long been viewed as the most challenging of classic games for artificial intelligence owing to its enormous search space and the difficulty of evaluating board positions and moves. Here we introduce a new approach to computer Go that uses 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. Without any lookahead search, the neural networks play Go at the level of state-of-the-art Monte Carlo tree search programs that simulate thousands of random games of self-play. We also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.

https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf

Deep learning style transfer: architecture

Deep learning allows easy combination architecture tasks.



$E_L = \sum (G^L - A^L)^2$
$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$
$G^L_{ij} = \sum_k F^L_{ik} F^L_{jk}.$
A^L
G^L
F^L
F^{L-1}
$\frac{\partial E_L}{\partial F^L}$
$\frac{\partial E_L}{\partial F^{L-1}}$
$\mathcal{L}_{content} = \sum (F^l - P^l)^2$
conv5_1 2 3 4
pool4
conv4_1 2 3 4
pool3
conv3_1 2 3 4
pool2
conv2_1 2
pool1
conv1_1 2
input
512
512
256
128
64
1 ...
# feature maps
$\mathcal{L}_{style} = \sum_l w_l E_l$
$\frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$
Gradient descent
$\vec{a} =$
$\vec{x} =$
$\vec{p} =$
$\vec{x} := \vec{x} - \lambda \frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$

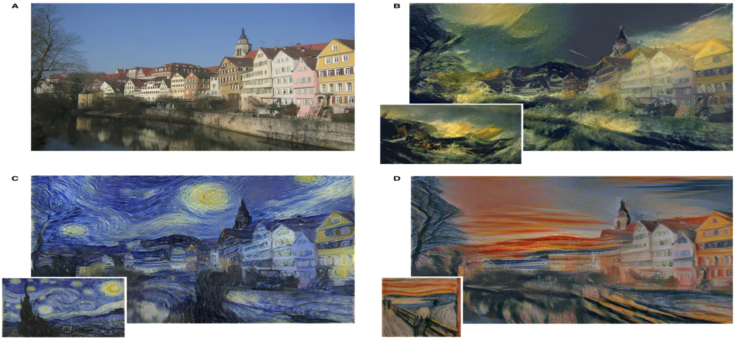
Figure 2. Style transfer algorithm. First content and style features are extracted and stored. The style image $\vec{a}$ is passed through the network and its style representation A^l on all layers included are computed and stored (left). The content image $\vec{p}$ is passed through the network and the content representation P^l in one layer is stored (right). Then a random white noise image $\vec{x}$ is passed through the network and its style features G^l and content features F^l are computed. On each layer included in the style representation, the element-wise mean squared difference between G^l and A^l is computed to give the style loss $\mathcal{L}_{style}$ (left). Also the mean squared difference between F^l and P^l is computed to give the content loss $\mathcal{L}_{content}$ (right). The total loss $\mathcal{L}_{total}$ is then a linear combination between the content and the style loss. Its derivative with respect to the pixel values can be computed using error back-propagation (middle). This gradient is used to iteratively update the image $\vec{x}$ until it simultaneously matches the style features of the style image $\vec{a}$ and the content features of the content image $\vec{p}$ (middle, bottom).

More complex architectures can allow learning and transferring characteristics of objects.

https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf

Deep learning style transfer: examples

Applications such as these seem impossible without DL



Photos can be transitioned to paintings with prescribed styles

`https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf`

Deep learning sound generation from video

DL is able to generate realistic synthetic data.

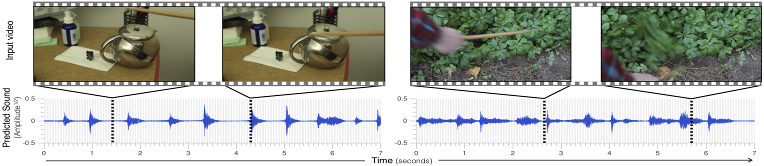


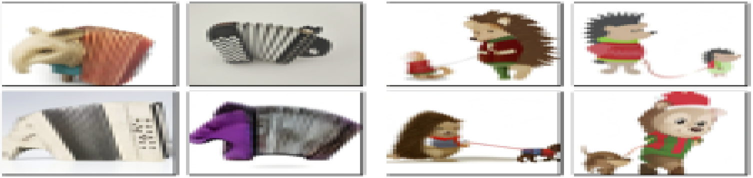
Figure 1: We train a model to synthesize plausible impact sounds from silent videos, a task that requires implicit knowledge of material properties and physical interactions. In each video, someone probes the scene with a drumstick, hitting and scratching different objects. We show frames from two videos and below them the predicted audio tracks. The locations of these sampled frames are indicated by the dotted lines on the audio track. The predicted audio tracks show seven seconds of sound, corresponding to multiple hits in the videos.

Expected sound characteristics can be learned from video with sound and then generated and added to video lacking sound.
https://arxiv.org/pdf/1512.08512.pdf
https://www.youtube.com/watch?v=0FW99AQmMc8
Many similar examples exist, see e.g. *DeepFakes* and automatic text generations. https://en.wikipedia.org/wiki/GPT-3

Deep learning image generation from text

Realistic images can be automatically generated from text

DALL·E: We've trained a neural network called DALL·E that creates images from text captions for a wide range of concepts expressible in natural language.



(a) a tapir made of accordion. a tapir with the texture of an accordion.

(b) an illustration of a baby hedgehog in a christmas sweater walking a dog

https://openai.com/blog/dall-e/

https://arxiv.org/abs/2102.12092

DeepLens for detecting rare events

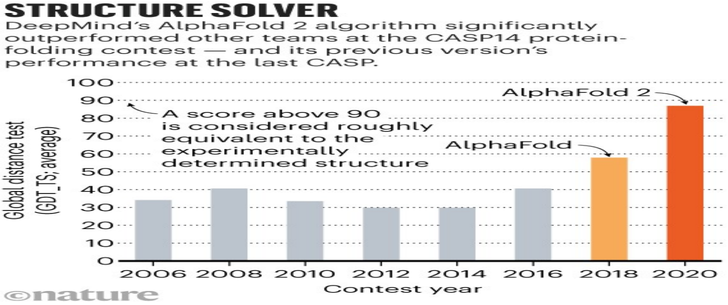
Some tasks are not scalable without machine learning.

For many science applications the quantity of data is beyond human inspection, use DL: Large Synoptic Survey Telescope `https://academic.oup.com/mnras/article/473/3/3895/3930852`

One approach to scale the visual inspection effort to the size of these surveys is to use crowdsourcing. This is the idea behind the Space Warps project (Marshall et al. 2015; More et al. 2015), which crowdsourced the visual inspection of a sample of 430 000 images from the CHFTLS to a crowd of 37 000 citizen scientists, yielding a new sample of gravitational lens candidates. The authors further estimate that a similar crowdsourcing effort can be scaled up to LSST sizes, where a considerable crowd of 10^6 *volunteers could visually inspect* 10^6 *LSST targets in a matter of weeks.*

AlphaFold

Predicting the 3D shape of proteins from its amino-acid sequence



STRUCTURE SOLVER

DeepMind's AlphaFold 2 algorithm significantly outperformed other teams at the CASP14 protein-folding contest — and its previous version's performance at the last CASP.

A score above 90 is considered roughly equivalent to the experimentally determined structure

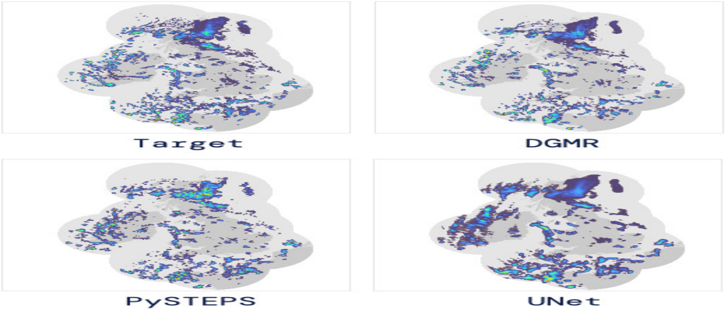
©nature

DNNs are increasingly used in scientific applications that historically required laborious lab work.

`https://www.nature.com/articles/d41586-020-03348-4`

Deep Generative Model for Rainfall (DGMR)

Predicting accurate near term rainfall on a fine mesh



Machine learning is increasingly state-of-the-art for scientific computing tasks.

`https://www.nature.com/articles/s41586-021-03854-z`

Machine Learning added mathematical conjectures

Aiding mathematicians proofs of fundamental conjectures

The practice of mathematics involves discovering patterns and using these to formulate and prove conjectures, resulting in theorems. Since the 1960s, mathematicians have used computers to assist in the discovery of patterns and formulation of conjectures1, most famously in the Birch and Swinnerton-Dyer conjecture2, a Millennium Prize Problem3. Here we provide examples of new fundamental results in pure mathematics that have been discovered with the assistance of machine learning—demonstrating a method by which machine learning can aid mathematicians in discovering new conjectures and theorems.

`https://www.nature.com/articles/s41586-021-04086-x`

Is your area of interest immune from deep learning?

Deep deep trouble: A case study by Michael Elad for image denoising:

Thousands of papers addressing this fundamental task [image denoising] were written over the years. Researchers developed beautiful and deep mathematical ideas with tools from partial differential equations, such as anisotropic diffusion and total variation, energy minimization viewpoint, adoption of a geometric interpretation of images as manifolds, use of the Beltrami flow, and more.... We have hence gained vast knowledge in image processing over the past three decades.

In 2012, Harold Burger, Christian Schuler, and Stefan Harmeling decided to throw deep learning into this problem. The idea was conceptually quite simple: take a huge set of clean images, add synthetic noise, and then feed them to the learning process that aims to turn a noisy image into its clean version. While the process was tedious, frustrating, and lengthy — the end result was a network that performed better than any known image denoising algorithm at that time.

`https://sinews.siam.org/Details-Page/deep-deep-trouble`

Maths of why DNNs: DNNs as function approximators

Functions act as classifiers and other machine learning tasks

Classification of inputs $x \in \mathbb{R}^n$ to c classes denoted by $\{e_i\}_{i=1}^{c}$, is modelled by a function $H(x)$ for which $H(x) = e_i$ for all x in class i where $e_i(\ell) = 1$ for $i = \ell$ and 0 otherwise.
Approximation Theory concerns the ability to approximate functions from a given representation; see accompanying lectures.

Some of the most well studied examples include approximation of a function $f(x)$ over $x \in [-1, 1]$ with some smoothness, say three times differentiable, by polynomials of degree at most k or trigonometric exponentials.

Here our focus is on the ability to approximate functions $H(x; \theta)$ given by a deep network architecture; for $x \in \mathbb{R}^n$.

Representational benefits of depth (Telgarsky 15')

Two layer ReLU network: sawtooth basis function

Telegarsky (2015) considered a specific construction of a function from a deep network which requires an shallow network to have exponential width.
Let $\phi(x) = ReLU(x) = max(x, 0)$ and consider the two layer net:

$$h_2(x) = 2\phi(x) - 4\phi(x - 1/2) = \begin{cases} 0 & x < 0 \\ 2x & x \in [0, 1/2] \\ 2 - 2x & x > 1/2 \end{cases}$$

and $h_3(x) = \phi(h_2(x))$ set to zero the negative portion for $x > 1$.
Here $W^{(1)} = (1\ \ 1)^T$, $b^{(1)} = (0\ \ -1/2)^T$, $W^{(2)} = (2\ \ -4)$, $b^{(2)} = 0$.
`https://arxiv.org/abs/1509.08101`

Representational benefits of depth (Telgarsky 15')

Composition gives exponential growth in complexity

For $\phi(x) = \max(x, 0)$ let $f(x) = h_3(x) = \phi(2\phi(x) - 4\phi(x - 1/2))$ and iterate this 2-layer network k times to obtain a $2k$-layer network $f^k(x) = f(f(\cdots(f(x)\cdots))$ with the property that it is piecewise affine with change in slope at $x_i = i2^{-k}$ for $i = 0, 1, \ldots, 2^k$ and moreover takes on the values $f^k(x_i) = 0$ for i even and $f^k(x_i) = 1$ for i odd.

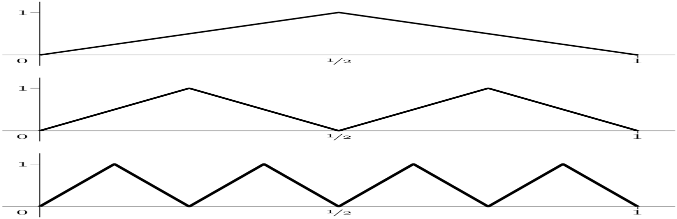


Figure 2: f_m, f_m^2, and f_m^3.

Representational benefits of depth (Telgarsky 15')

Composition gives exponential growth in complexity: width vs. depth

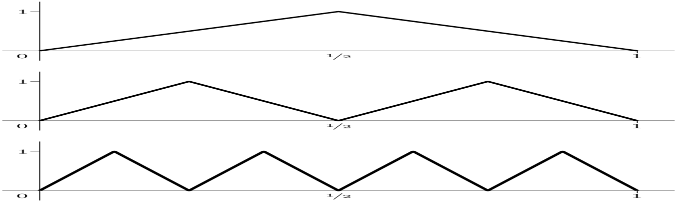


Figure 2: f_{m}, f_{m}^2, and f_{m}^3.

In contrast, a two-layer network with the same $\phi(x)$ of the form $\phi\left(\sum_{j=1}^{m} \alpha_j \phi(w_j x - b_j)\right)$ requires $m = 2^k$ to exactly express $f^k(x)$.

The deep network can be thought of as having $6k$ parameters, whereas the two-layer network requires $3 \cdot 2^k$ parameters; exponentially more. `https://arxiv.org/abs/1509.08101`

Representational benefits of depth (Telgarsky 15')

Classification error rates

Define the function class $F(\phi; m, \ell)$ be the space of functions composed of ℓ layer fully connected m width feed forward nets with nonlinear activation function ϕ. Let $\mathcal{R}(f) := n^{-1} \sum_{i=1}^{n} \chi[f(x_i) \neq y_i]$ count the number of incorrect labels of the data set $\{(x_i, y_i)\}_{i=1}^{n}$.

Theorem 1.1 (Telgarsky 15')

Consider positive integers k, ℓ, m with $m \leq 2^{(k-3)/\ell-1}$, then there exists a collection of $n = 2^k$ points $\{(x_i, y_i)\}_{i=1}^{n}$ with $x_i \in [0,1]$ and $y_i \in \{0,1\}$ such that

$$\min_{f \in F(\phi;2,2k)} \mathcal{R}(f) = 0 \quad \text{and} \quad \min_{g \in F(\phi;m,\ell)} \mathcal{R}(g) \geq \frac{1}{6}.$$

https://arxiv.org/abs/1509.08101

Representational benefits of depth (Telgarsky 15')

More general bound.

Let $\mathcal{F}(\phi; m, \ell; k)$ denote space of functions generated by composing functions from $\mathcal{F}(\phi; m, \ell)$ k times.

Theorem 1.2 (Telgarsky 15')

Consider positive integers k, number of layers ℓ with m width per layer. Let $\phi(\cdot)$ be a $t-$sawtooth function and $\phi_R(x) = \max(x, 0)$ is a $2-$sawtooth function. Consider the data $x_i = i2^{-k}$ for $i = 0, 1, \ldots, 2^k$ with $y_i = 0$ for i even and $y_i = 1$ for i odd, then

$$\min_{f \in F(\phi_R; 2,2;k)} \mathcal{R}(f) = 0 \quad \text{and} \quad \min_{g \in F(\phi; m, \ell)} \mathcal{R}(g) \geq \frac{n - 4(tm)^{\ell}}{3n}.$$

Theorem 1.1 follows from Theorem 1.2 with $m \leq 2^{(k-3)/\ell - 1}$.

`https://arxiv.org/abs/1509.08101`

Representational benefits of depth (Telgarsky 15')

Ingredients of the proof

We refer to $\phi(\cdot)$ as a $t-$ sawtooth function if it is piecewise affine with t pieces; that is $\mathbb{R}$ is partitioned into t consecutive intervals and $\phi(\cdot)$ is affine within each interval.

Lemmas 2.1(Telgarsky 15')

Let $\phi(\cdot)$ be a $t-$sawtooth function then every $f \in \mathcal{F}(\phi; m, \ell)$ with $f : \mathbb{R} \to \mathbb{R}$ is a $(tm)^{\ell}$-sawtooth.

Lemmas 2.3(Telgarsky 15')

Let $f : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ be $k-$ and $\ell-$sawtooth functions respectively, then $f + g$ is a $(k + \ell)-$sawtooth function and $f \circ g$ is a $k\ell-$sawtooth function.

https://arxiv.org/abs/1509.08101

Representational benefits of depth (Telgarsky 15')

Proof of Theorem 2.1: $g \in F(\phi; m, \ell)$ at $(mt)^{\ell}-$sawtooth functions

Consider the first layer $h_2 = \phi(Wx + b)$, each entry of this m-vector is an affine function, i.e. $1-$sawtooth, for which ϕ is then applied and as it is a $t-$sawtooth each entry in $\phi(Wx + b)$ is a $t-$sawtooth.

The second layer has each entry in $Wh_2 + b$ being a liner combination of m different t-sawtooth functions, resulting in each entry of $Wh_2 + b$ being a mt-sawtooth function. The second layer concludes with $h_3 = \phi(Wh_2 + b)$ which is composing a $t-$sawtooth function with each mt-sawtooth function for each entry in h_3 being an mt^2-sawtooth function.

Inductively $h_\ell \in \mathbb{R}^m$ has as each entry $m^{-1}(mt)^i$-sawtooth functions and the final from $\mathbb{R}^m$ to $\mathbb{R}$ is a linear combination of these m functions to give an $(mt)^{\ell}-$sawtooth function.

`https://arxiv.org/abs/1509.08101`