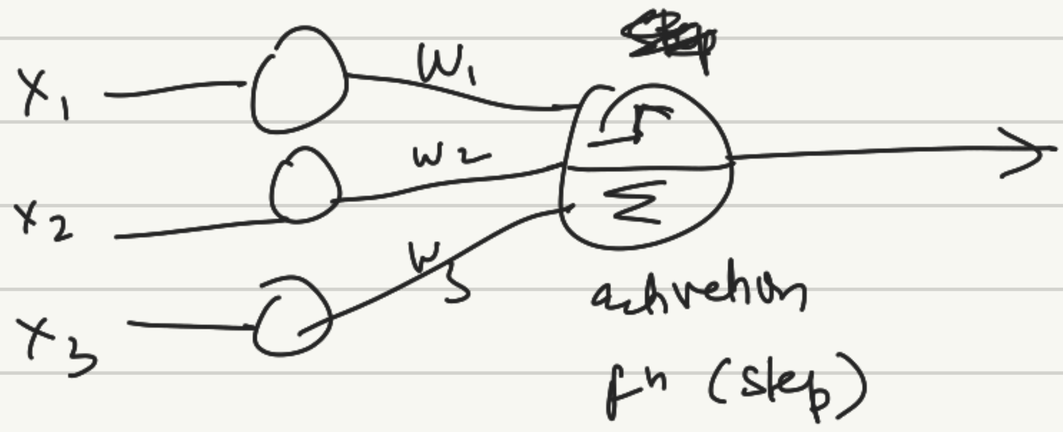
Dec 29 - 2025

ANN - Artificial neural net

TLU



$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \qquad w = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

Training a perceptron

Hebb's rule

learning rate

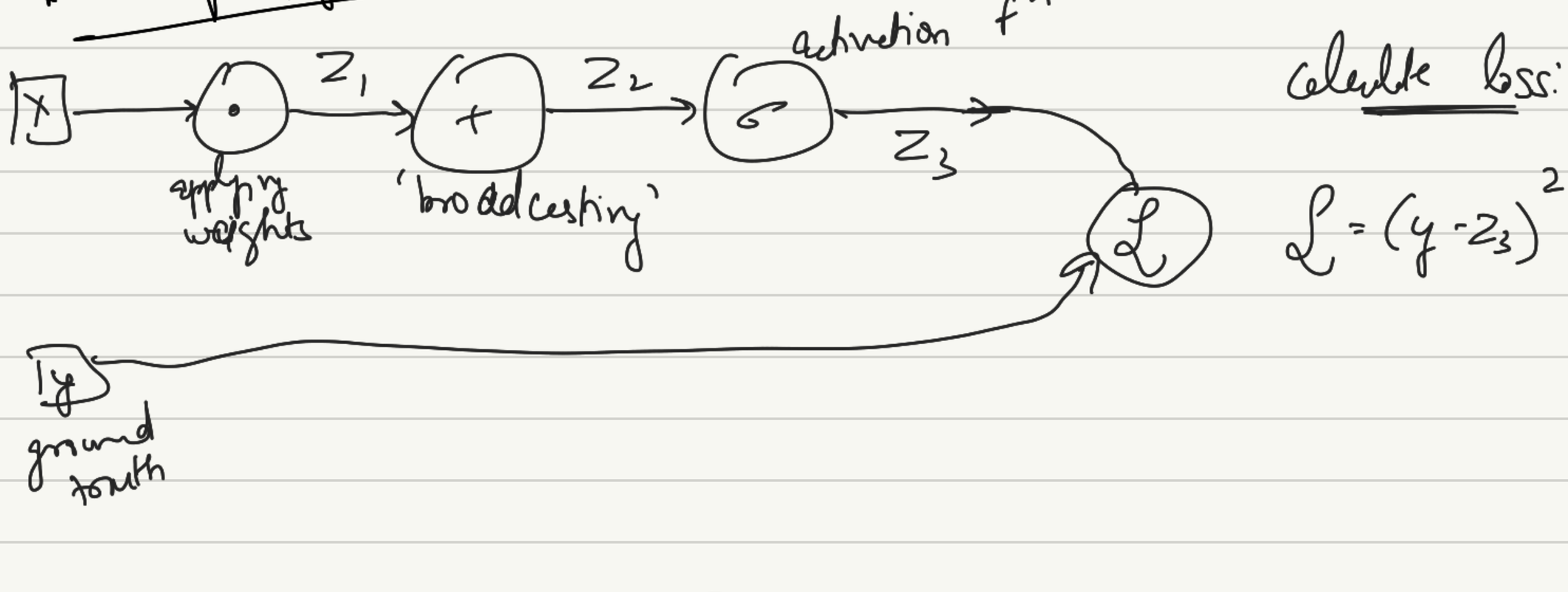
$$w_{ij}^{(\text{next step})} = w_{ij} + \eta (y_i - \hat{y}_j) x_i$$

loss fn: $L = (y - z_3)^2$

y ← ground truth

$\hat{y}$ ← ML predicted

Back propagation



calculate loss:

backpropagation

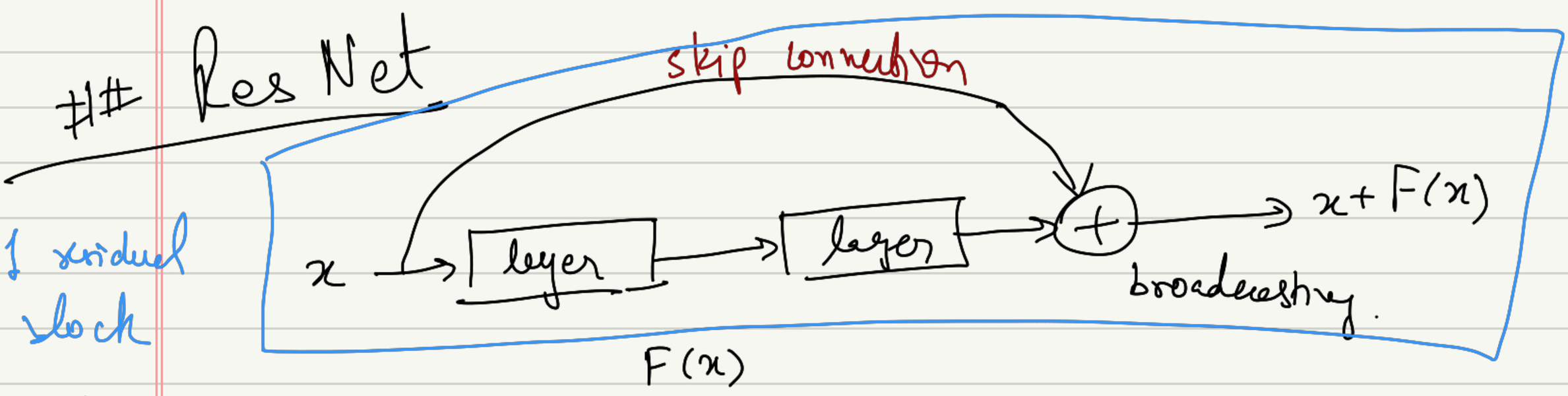
$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial z_3} \cdot \frac{\partial z_3}{\partial z_2} \cdot \frac{\partial z_2}{\partial z_1} \cdot \frac{\partial z_1}{\partial W}$$

back propagation

$$\frac{\partial L}{\partial b} = \left(\quad \right) \frac{\partial z_1}{\partial b}$$

- ReLU activation fn $\quad f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$

ResNet



1 residual block

→ proposed to solve the problem of vanishing / exploding gradients

ResNet is made up of multiple residual blocks.

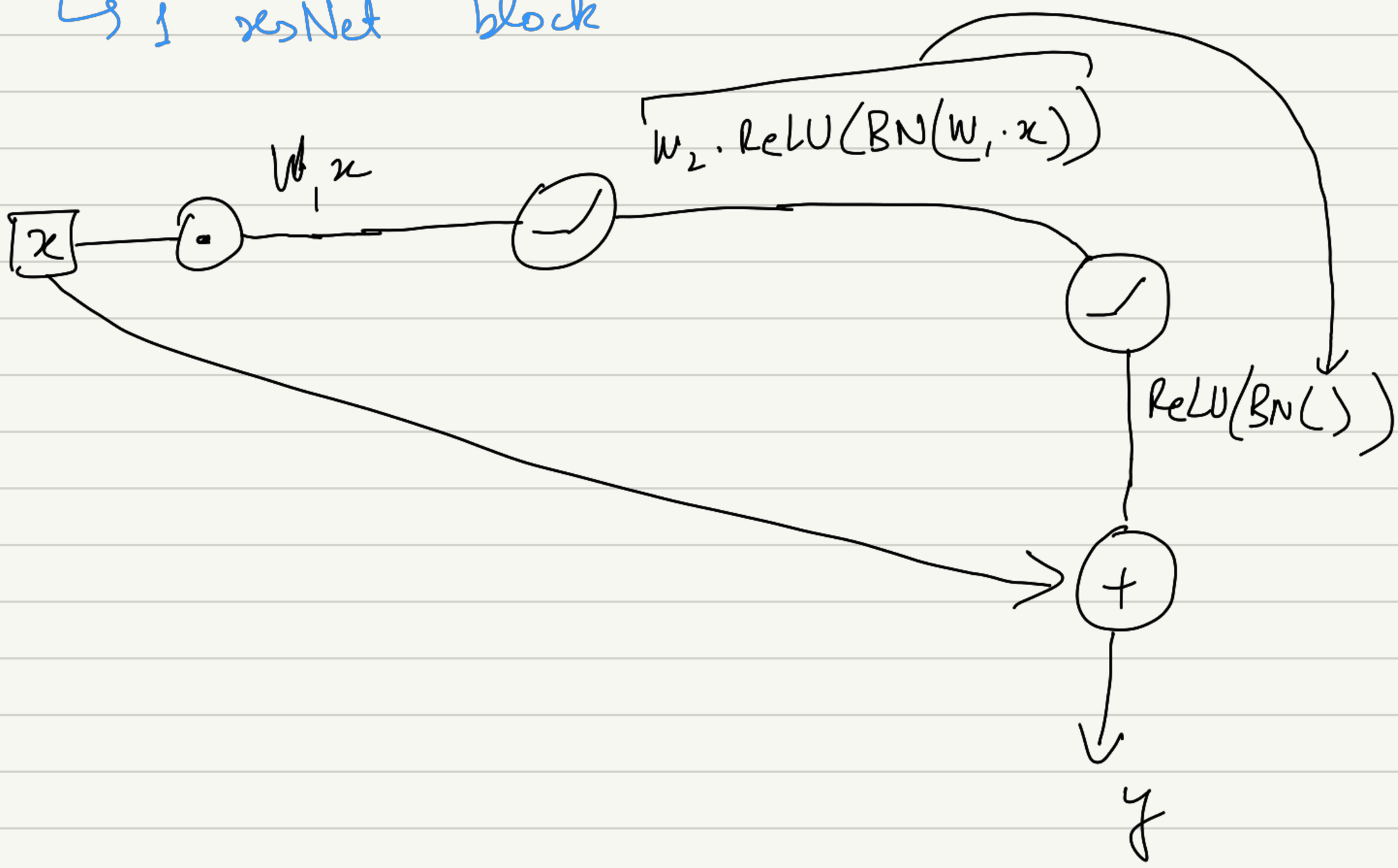
→ Regularization??

ASE - ResNet (Laydi et. al. 2025)

Encoding path

$$y = \mathrm{ReLU}\left(\mathrm{BN}\left(W_2 \cdot \mathrm{ReLU}\left(\mathrm{BN}\left(W_1 \cdot x\right)\right)\right) + x\right)$$

↳ 1 resNet block



- there's 4 such residual blocks.
- this is standard resnet, nothing new here

Decoding path

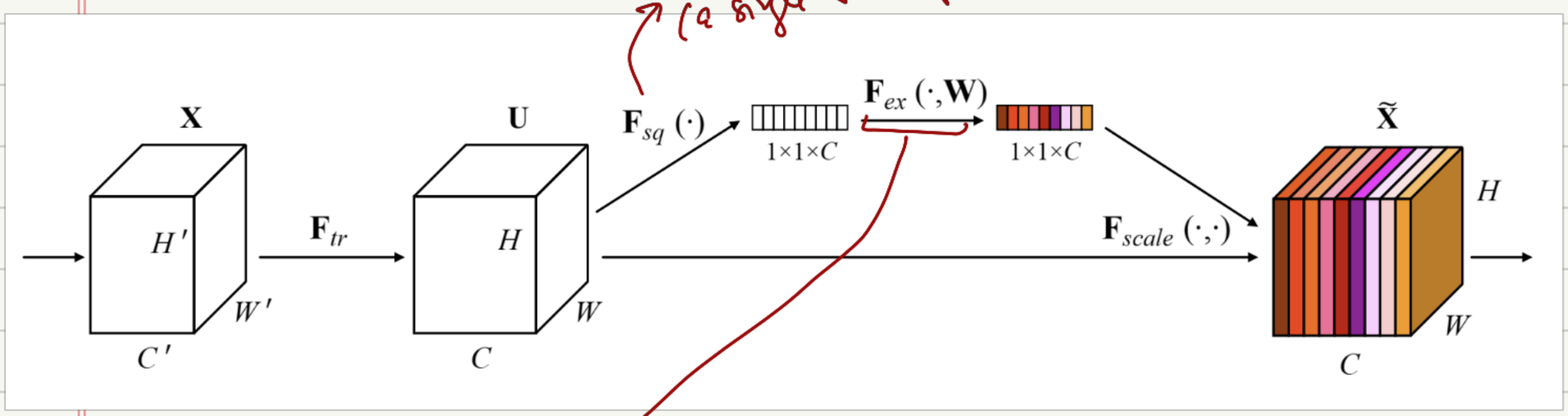
- 4 upsampling blocks.

. . - - . →

Main idea for "Squeeze & exciteion"

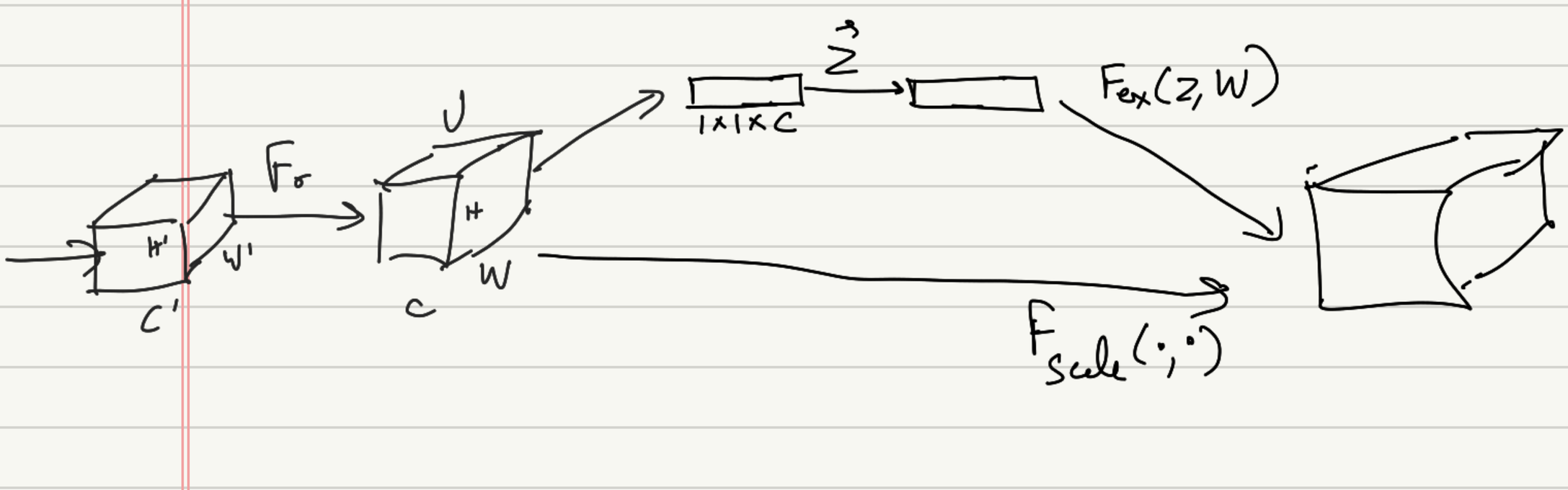
Channel wise "attention" mechenism.
OR, "attention" is focused on the per channel information.

↳ information in some channels is more important
(ex: the 'Green' channel would have more information in an image of the grass)



excitation operator, to figure out (or assign?) the importance of each channel.

(SEnet, Hu et. al. 2018)



<... ## Decoding path

- noise level: $N = \sigma(\text{conv2d}(x))$, $N \in R^{B \times H \times W}$

  ↳ global average pooling: $\hat{N} = \frac{1}{B \times H \times W} \sum_{b=1}^{B} \sum_{i=1}^{H} \sum_{j=1}^{W} N_{b,i,j}$

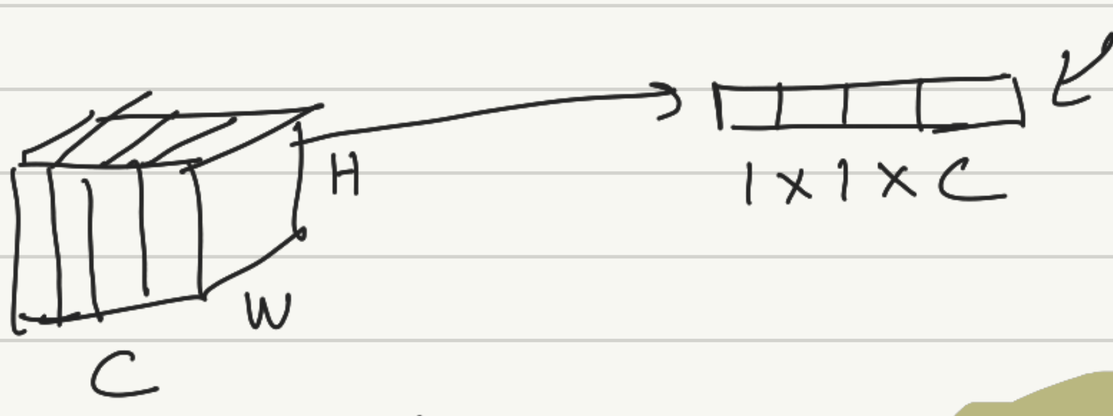
- $\gamma_{adaptive}$: $\max(1, \lfloor 16 \cdot (1 + \hat{N}) \rfloor)$

  ↳ reduction ratio

Dec 30

Bottleneck & reduction ratio:

In M2, bottleneck is a compression point



Reduction ratio: $\gamma = \frac{\text{Total channels}}{\text{bottleneck size}}$

e.g ~~for 64 channels in an~~ an image with 64 channels & a reduction ratio 16, would have:

$$\text{Bottleneck size} = \frac{\text{total channels}}{\gamma} = \frac{64}{16} = 4 \text{ neurons.}$$

an industry standard to have $\gamma = 16$ but can be any other value depending on the problem.

*
- regular SENet : $r = 16$
- adaptive SENet : $r_{adaptive} = \max(1, 16[1+\hat{N}])$

Why does this expression work? What's 'adaptive' about it?

→ $\hat{N}$ (noise level) is measured for each channel

↳ low noise ($\hat{N} \approx 0$) : $r \approx 16$ which will lead to standard bottleneck size.

↳ high noise ($\hat{N} \approx 1$):

$\Rightarrow r = 16(1+\hat{N}) \simeq 16(1+1)$

$\Rightarrow \boxed{r \times 32}$

↳ higher reduction ratio for higher noise values

* r is updated on the fly corresponding to the amount of noise per channel.

* the $\max(1, \ldots)$ is to ensure the reduction ratio is never zero.

... →

Concatenation of feature maps

for two feature maps with following dimensions:

$$H \times W \times C_1 \quad , \quad H \times W \times C_2$$

concatenation will produce a feature map with dimensions:

$$H \times W \times (C_1 + C_2)$$

for concatenation,

- two out of three dimensions must be the same
  - the one that is not the same (usually the channel dimension) is the axis of concatenation.

Loss fn's used in Leydi et. al. 2025

1.) Multiclass cross entropy loss (

from PHY 391 Note 3 supplement:

4. Cross-entropy in information theory/ML

Suppose p_i = true probability distribution

q_i = predicted probability distribution (e.g. one we derive by a training procedure in ML)

Cross-entropy is (log-loss): $H(p,q) = -\sum_i p_i \log_2 q_i$

The minimum value of cross-entropy is when $q_i = p_i$ ("Gibbs' inequality" result)

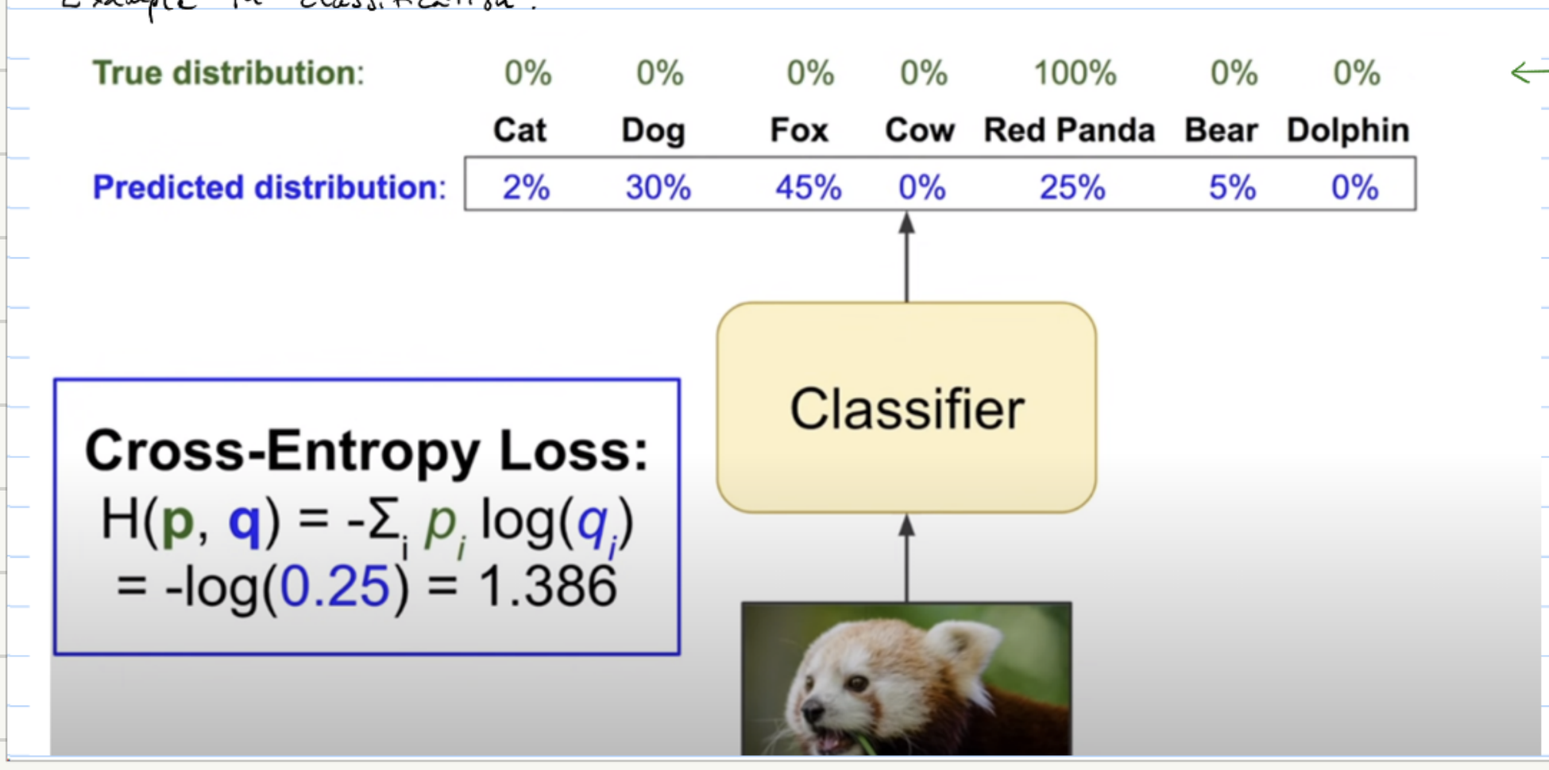
⇒ minimum value of $H(p,q)$ is entropy $H(p) = -\sum_i p_i \log_2 p_i$

The difference between entropy and cross-entropy is the Kullback-Leibler (KL) divergence

$D_{KL}(p\|q) = H(p,q) - H(p)$

Thus, using cross-entropy as loss function in ML we try to adjust q_i close to p_i

Example in classification:



Note entropy of p_i distribution is zero! p_i is either zero or one

Dec 31

Jan 1

Jan 2

~~Syllabus good~~

Single Gaussian Covariance matrix

	A	μ	σ
A	var(A)		
μ		var(μ)	
σ			var(σ)

Double Gaussian covariance matrix

	A	μ_1	σ_1	B	μ_2	σ_2
A	var(A)			Cov(A,B)		0
μ_1						1
σ_1						2
B	cov(A,B)			var(B)		3
μ_2						4
σ_2						5
	0	1	2	3	4	5

- by default, scipy uses Levenberg-Marquardt algorithm when bounds are not specified (source: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html#:~:text=curve_fit%20is%20for%20local%20optimization,%2Db%20*%20x)%20+%20c)

Gradient descent when far from optimal solution

Gauss-Newton when close to the minimum.

— x —

Jan 3

Revisiting thermodynamic potentials & phase transitions.

Phase transitions

abrupt

smooth

* 1st order transition ⇒ a discontinuous transition.

Gibbs Enthalpy

$$G = U(S,V,N) - TS + PV$$

$$dG = -SdT + VdP + \mu dN$$

$$\left(\frac{\partial G}{\partial T}\right)_{P,V} = -S + 0 + 0$$

$$V = \left(\frac{\partial G}{\partial P}\right)_T$$

Jan 4

Jan 5

#Jan 6

#Jan -7

1 particle ~~state~~ in 1-D → state.size() = 3

state [0] : x rate [0] = $\dot{x}$

state [1] : V_x rate [1] = $\dot{V}_x$

state [2] : ~~t~~ rate [2] = $\dot{t} = \frac{dt}{dt} = 1$

this is useful to code system of ODEs to time evolve the system. For example:

Velocity verlet algo:

$$x_{n+1} = x_n + V_n \Delta t + \frac{1}{2} a_n (\Delta t)^2$$

$$V_{n+1} = V_n + \frac{1}{2} (a_{n+1} + a_n) \Delta t$$

to code these, we can write:

state [i] = state [i] + state [2*i] * dt

$$+ \frac{1}{2} \text{rate}[2*i] * (dt)^2$$

state [2*i] = state [2*i] + (1/2)(

the EOM of a pendulum is given by:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L} \sin\theta$$

also, $V = L\frac{d\theta}{dt}$, $a = \frac{d^2\theta}{dt^2} = \frac{dV}{dt} = -\frac{g}{L}\sin\theta$

state [0] → θ rate [0] → $d\theta/dt = V$ = state [1]

state [1] → V rate [1] → $\frac{dV}{dt} = a = -\frac{g}{L}\sin(\text{state}[0])$

state [2] → t rate [2] → $\frac{dt}{dt} = 1$

For a Bouncing Ball

EoM:

$$ma = -mg$$

$$\Rightarrow a = -g$$

state[0] → y rate[0] → $\frac{dy}{dt}$ = state[1]

state[1] → $\frac{dy}{dt}$ → v rate[1] → $\frac{dv}{dt}$ = a = −g

state[2] → t rate[2] → 1

Legendre transforms

- f^n of a variable → (Legendre transform) → f^n of conjugate variable

⇒ both f^n's have same units

e.g.1 L & H both have units of energy

e.g.2 : U, H, F and G (thermodynamic potentials) have unit of energy.

(here the conjugate pair of variables are pressure P, volume V, temperature T & entropy S)

* [redacted] conjug

- How are conjugate variables defined?

→ from 'Uncertainty principle':

$$\Delta x \, \Delta p \geq \hbar/2$$

∴ x & p are conjugate variable pairs

* Mathematically x & p follow Uncertainty principle b/c they are conjugate variables!!

⇒ In thermodynamics, the product of conjugate variables have units of energy.

↳ Work = force × displacement

↓

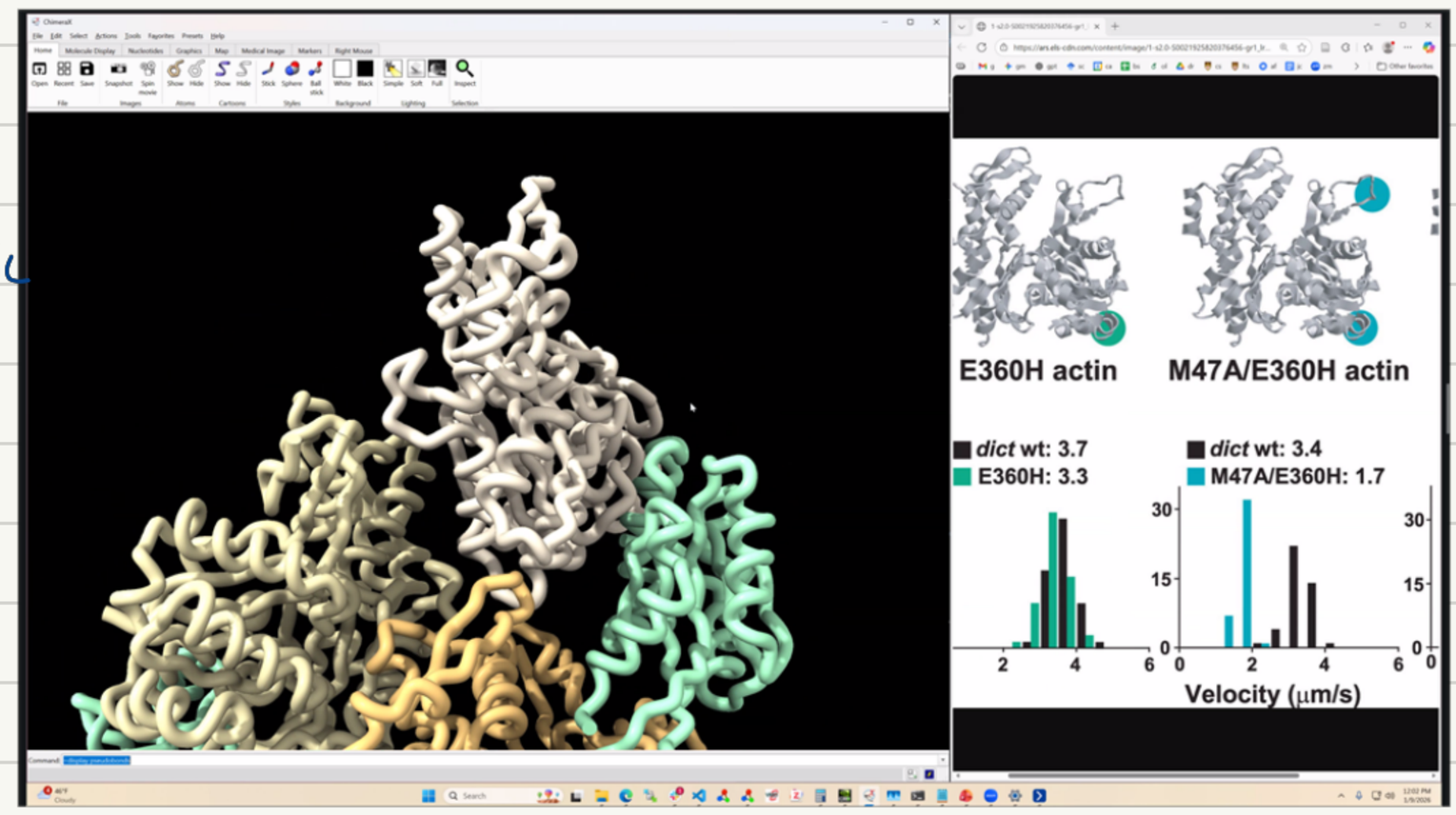
this holds for any generalized force (Pressure & temp. in thermodynamics) & generalized displacement (like volume & entropy in thermodynamics)

* So as a rule of thumb, a pair of generalized force & generalized velocity makes a conjugate variable pair.

- D loop of actin monomer
  - ↳ cofilin breaks the D-loop connection on
  - ↳ on two actins.

* site of hydrolysis on actin.
  - ↳ monomeric actin structure.

* groove portion of the hydrophobic plaque



→ thoronin
  - ↳ another severing protein like cofilin

→ ~~pointed~~ pointed end polymerization.
  - ↳ hydrolysis
    - ↳ detailed balance
      - ↳ Prof. Vavylonis thought about this for a year.

= plot counts histogram.] over time
↳ more lightness.

⇒ log - log plot.
↳ to include the outliers

→ ~~to~~ remove spikes and analyze
↳ don't have to keep verifying
↳ another graph.

↳ bias vs precision etc.

⇒ ~~Summary~~ for Dipl

- ↳ get back to Basel