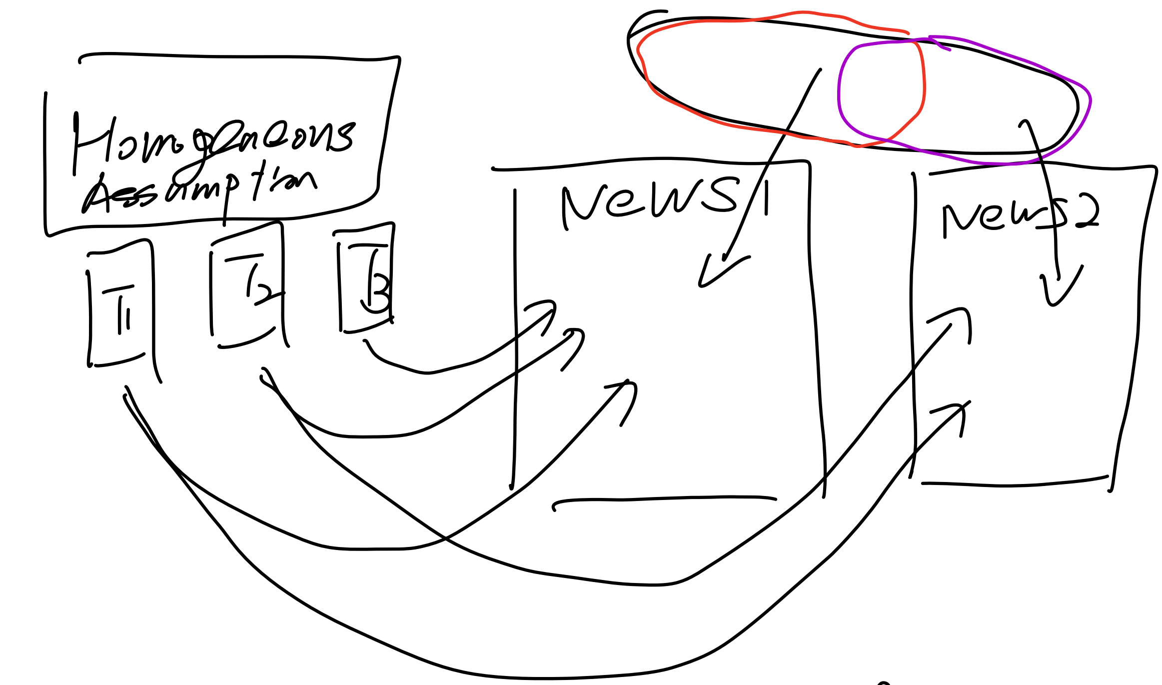
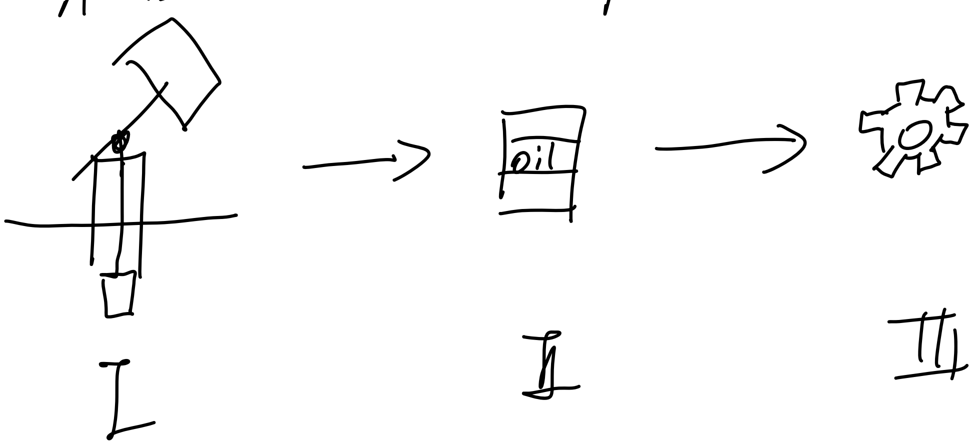


Bayesian classifier:

Need to Classify Good & Bad for inc A.

A is a crude oil production inc.



Text I

oil has release
new tech on
oil extraction

Text II

WB bought $100M to
oil Inc. ...

making it fast
and high-quality

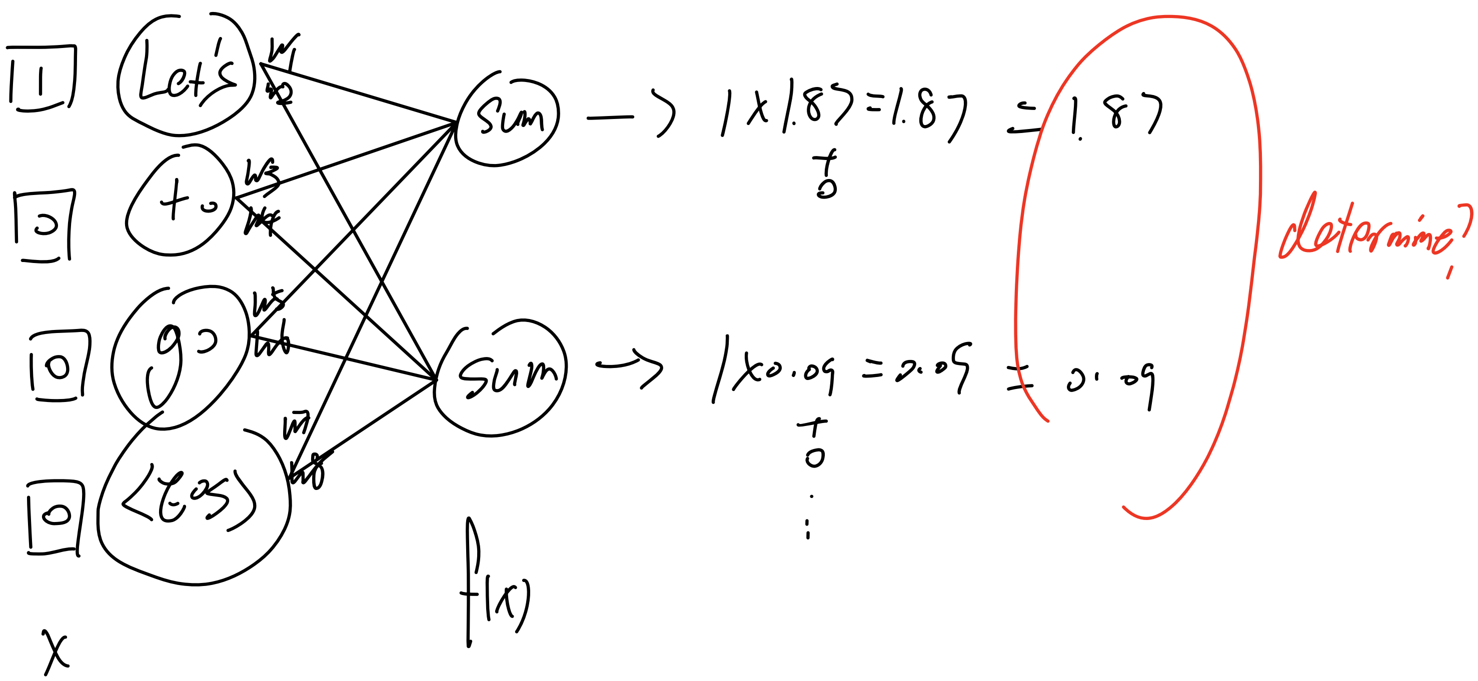
⇓

☆ Text Detection (NLP)

Transformer

① Word Embedding

$\boxed{\text{Text} \rightarrow \text{Number}}$



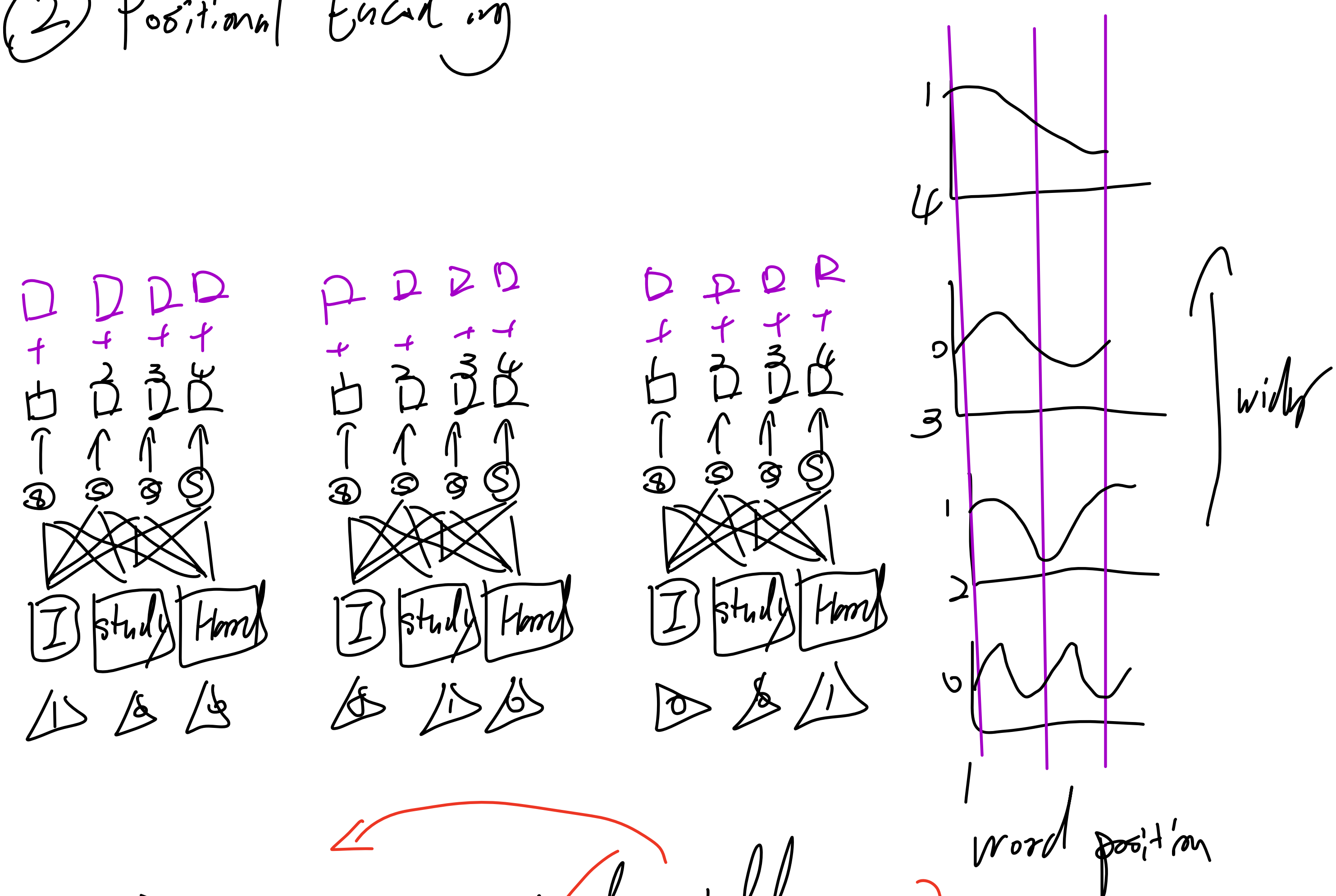
Let's = (1.87, 0.09)

↑ same weights

go = (-0.78, 0.27)

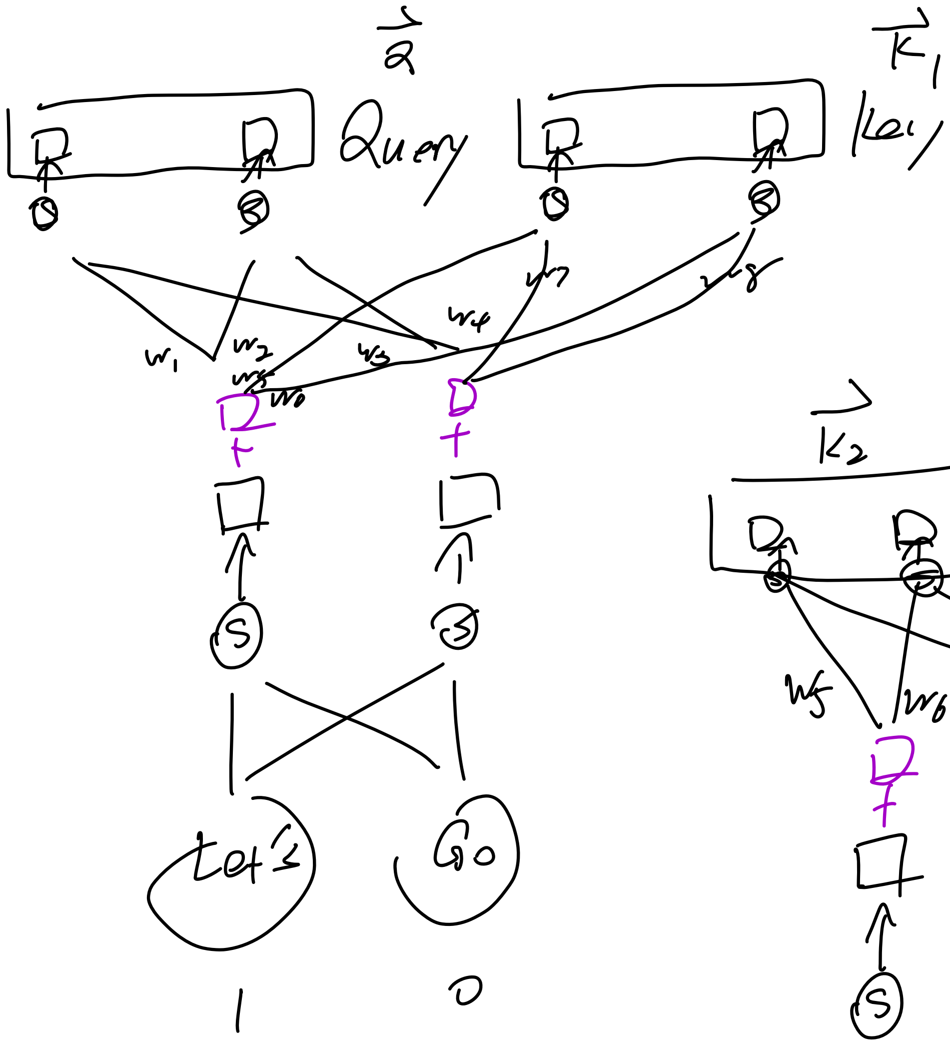
weights determined by Backpropagation

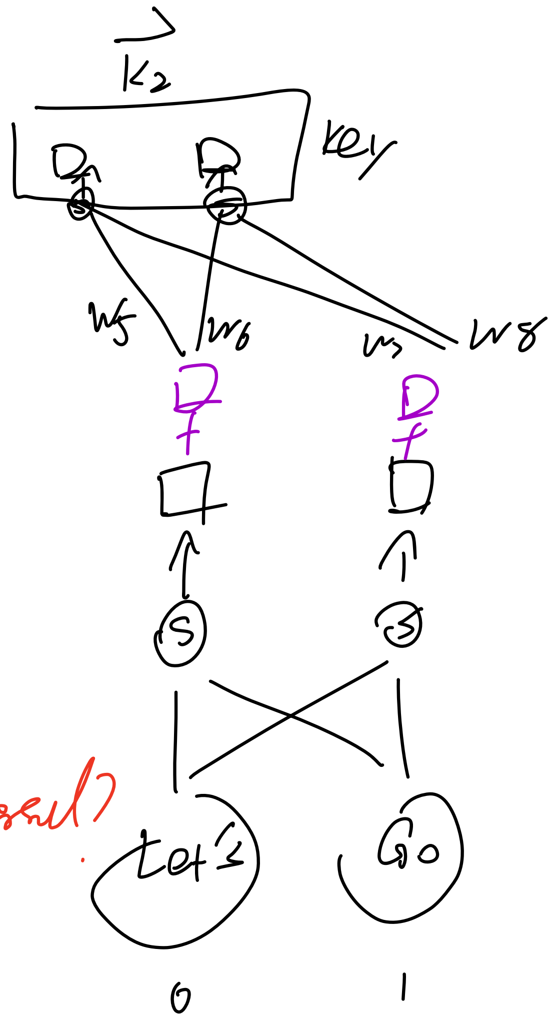
② Positional Encoding



=> unique as word embedd more
+ wider

③ Self-Attention





How but not Reversed?

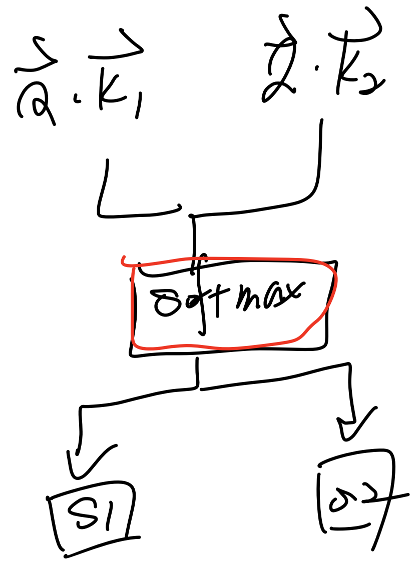
$\vec{q}\cdot\vec{k}_1$ $\quad$ $\vec{q}\cdot\vec{k}_2$

$\parallel$ $\quad$ $\parallel$

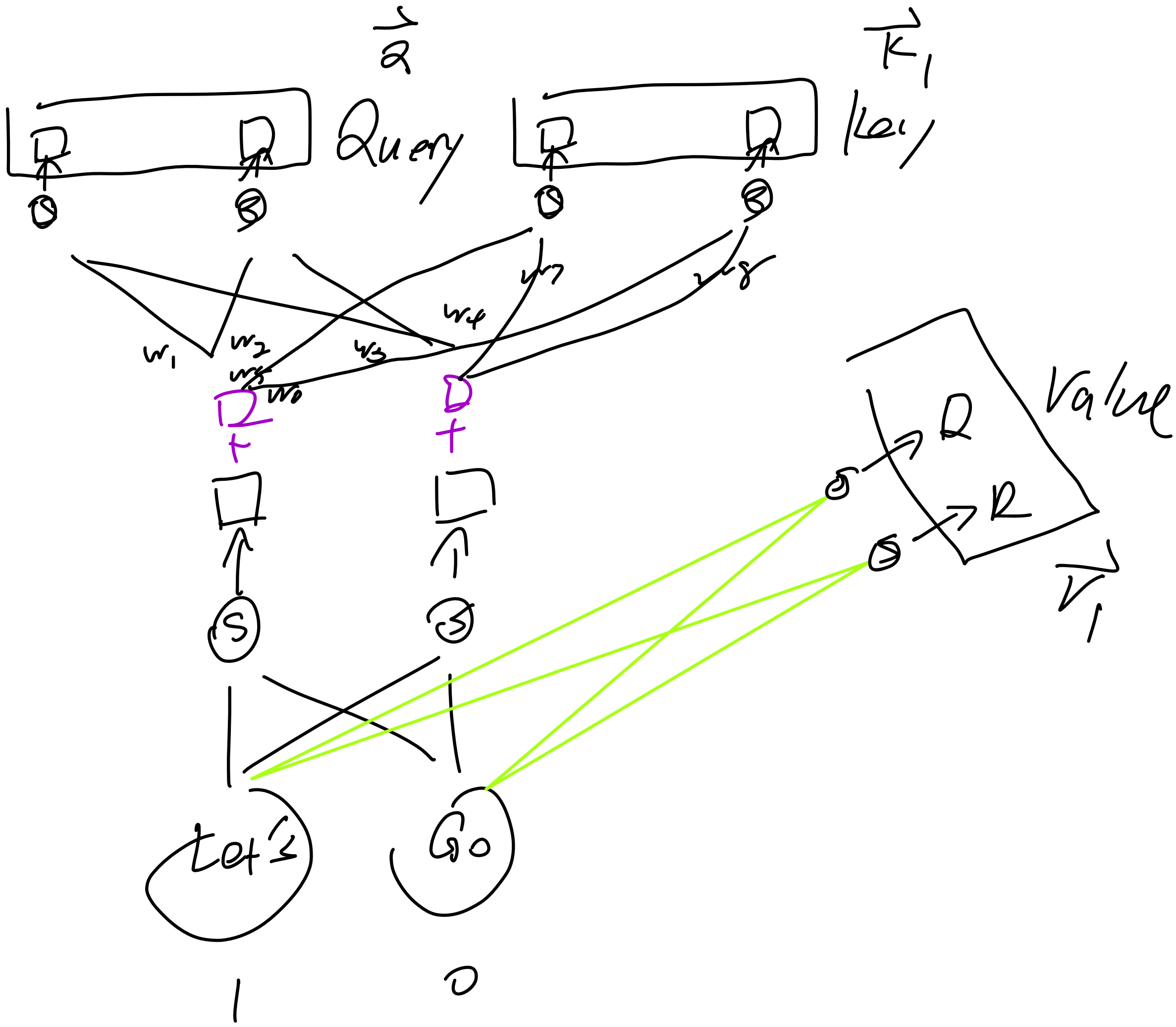
11.7 $\quad$ −2.6

⇒ $\boxed{\text{Let's} \;\infty\; \text{Let's}}$

I sell the stock since it shows bad performance



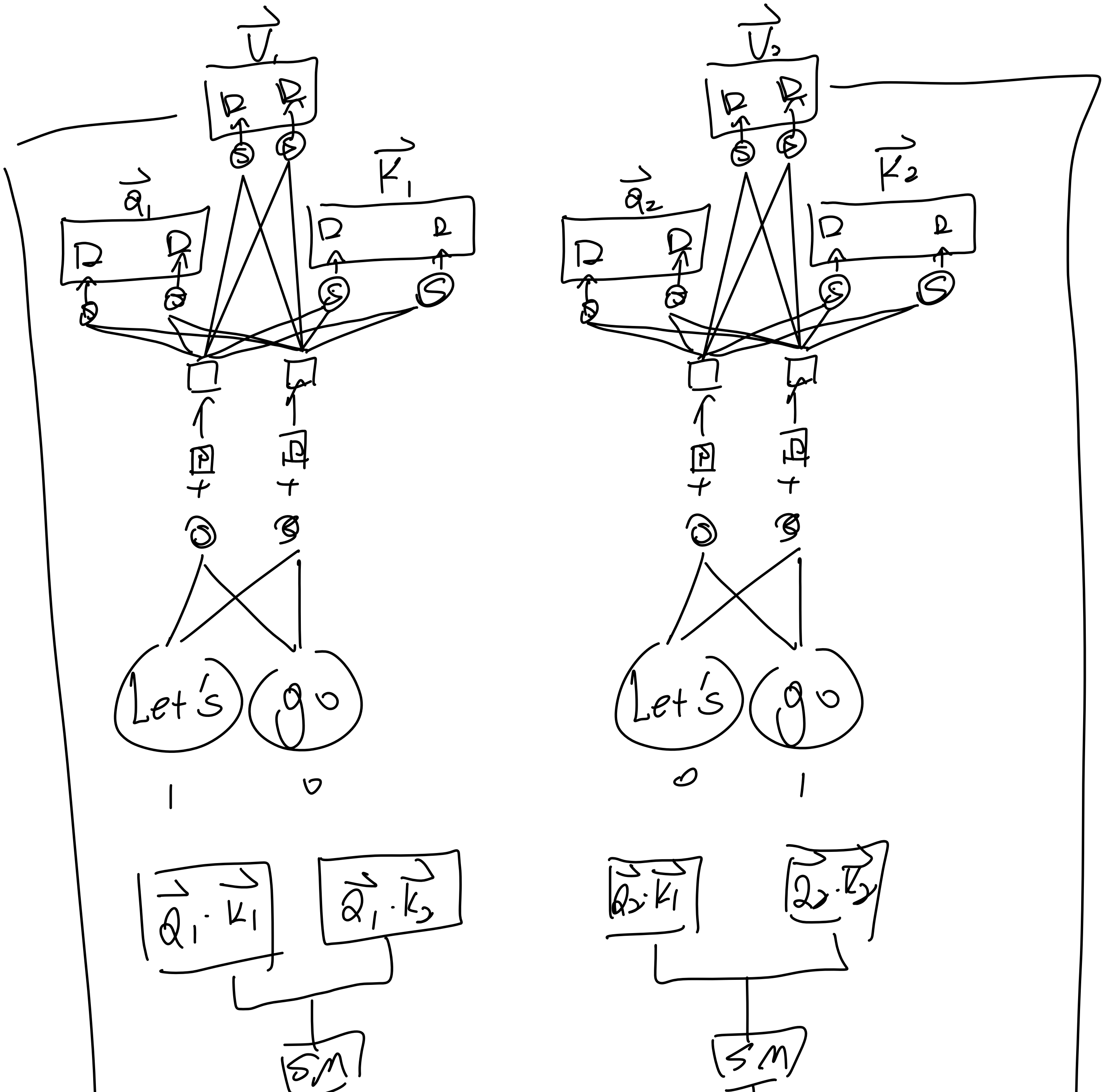
Q · K1
Q · K2
softmax
S1
S2



Q
K1
Query
Key
w1
w2
w3
w4
Value
V1
Let's
Go
1
0

$$\vec{V}_{11}\cdot S1 \qquad \vec{V}_{12}\cdot S_1$$
$$+\vec{V}_{21}\cdot S_2 \qquad \vec{V}_{22}\cdot S_2$$
$$= S_1^* \qquad = S_2^*$$

= self-attension value for Let's



$\boxed{S_1}$ $\boxed{S_2}$

$$V_{11}\cdot S_1 + V_{21}\cdot S_2 = SA_1$$

$$V_{12} S_1 + V_{22}\cdot S_2 = SA_2$$

$$\begin{bmatrix} V_{11} & V_{21} \\ V_{12} & V_{22} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} = \begin{bmatrix} SA_1 \\ SA_2 \end{bmatrix}$$

$\vec{V_1}$ $\vec{V_2}$

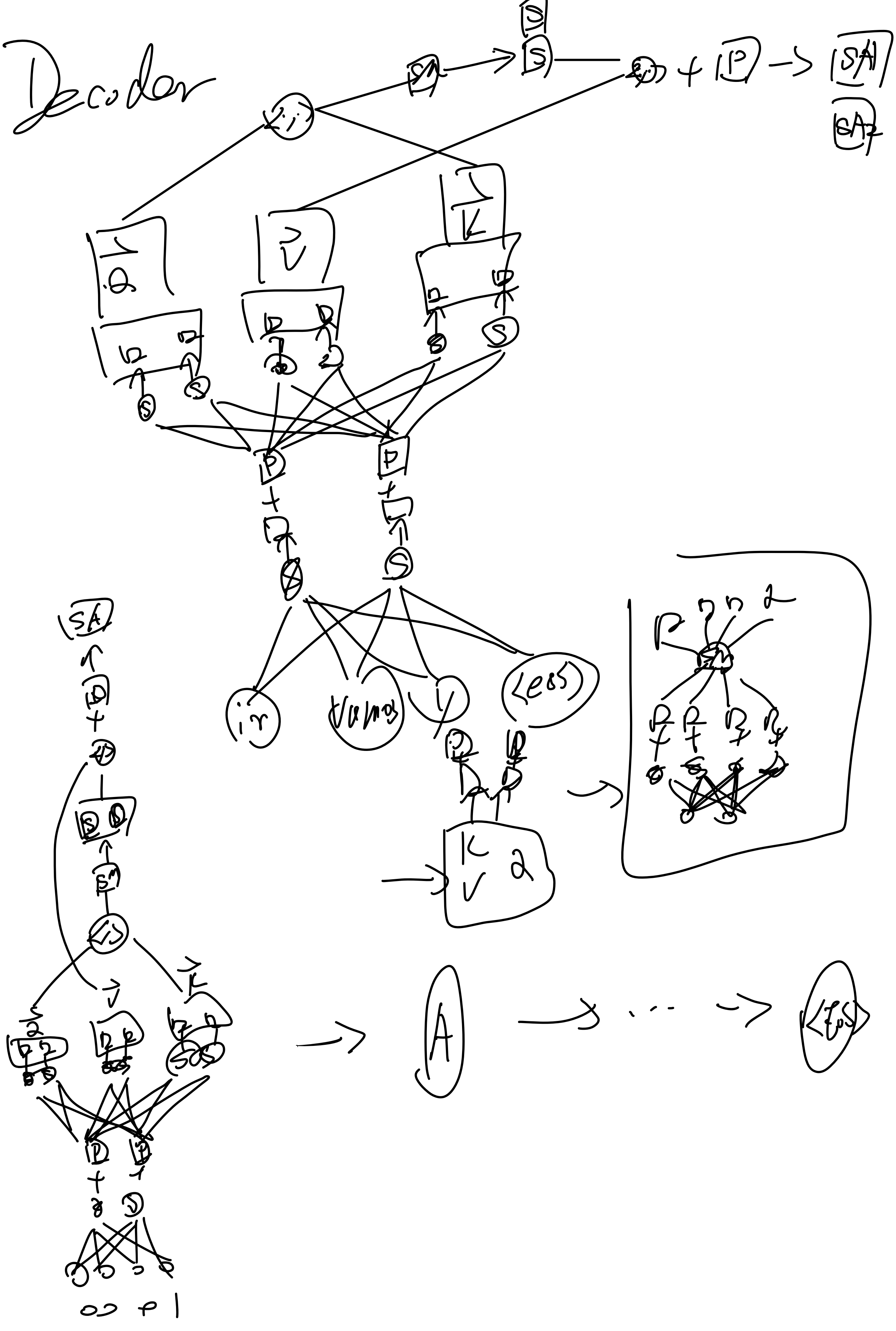
$\boxed{SA_1}$ + $\boxed{P}$ → $\boxed{PE_1}$

$\boxed{SA_2}$ + $\boxed{P}$ → $\boxed{PE_2}$

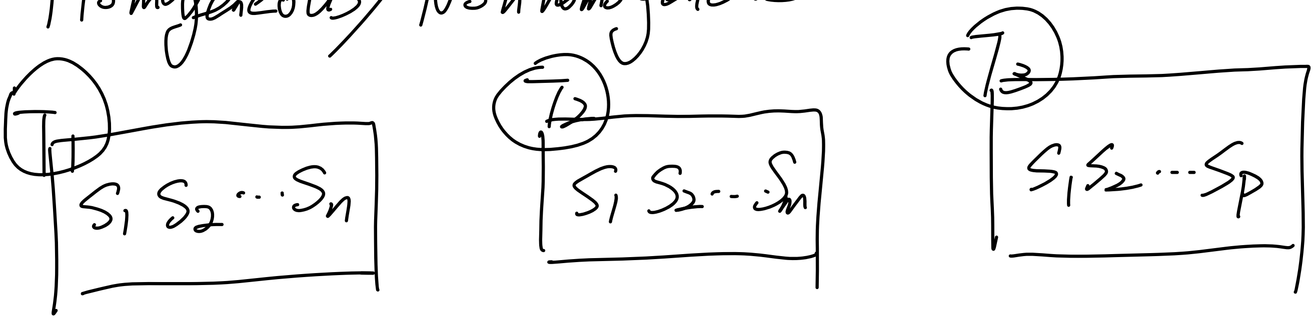
Word Embedding → Positional Encoding → Self-Attention → Residual Connection

Encoder

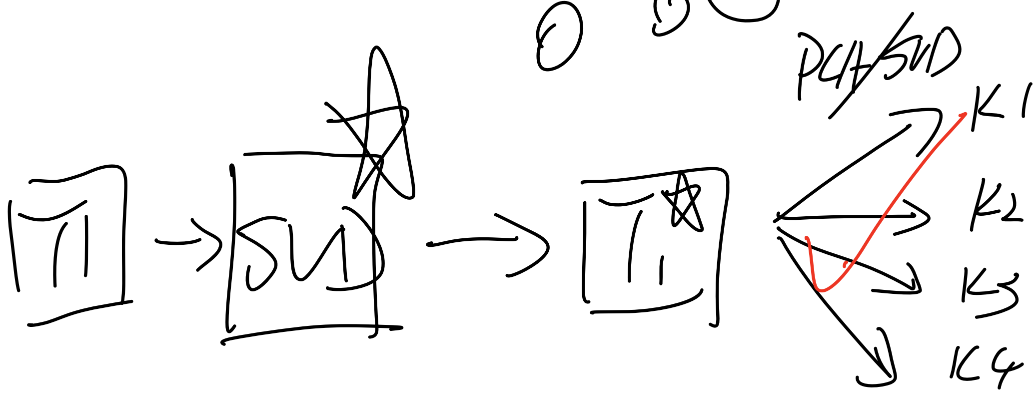
Decoder

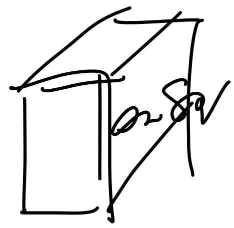


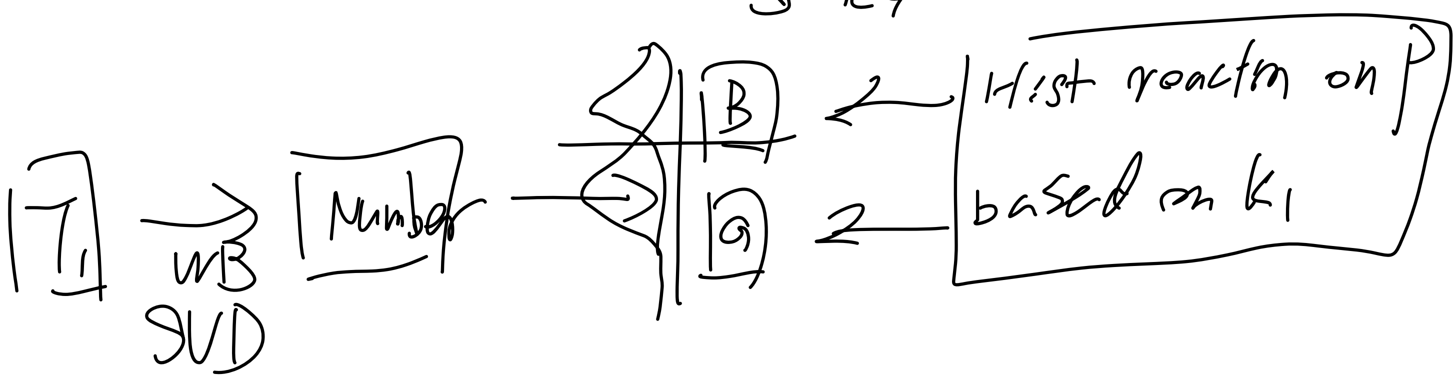
Homogeneous/Nonhomogeneous



one-hot word2vec







to make sure stochastically uniform in t

$P(t, K_1, K_2)$

Why Res min?

Homogeneous/Nonhomogeneous

Define $T: S \longrightarrow V$

S is sentense space

V is normal vector space (Euclidian Norm)

Then $T(s_1) = v_1 \cdots T(s_n) = v_n$

T is the same as Word Embeding

so $T(\cdot) \Leftrightarrow WB(\cdot)$

Similarly,

we have $P(\cdot)$, $SA(\cdot)$ as positional Encoding and Self-Attention

$$\Rightarrow \ TEncoder(\cdot) = SA \circ P \circ T^n$$

similarity among words

order infor

WB

Note $\boxed{\text{Word2vec}}$ can be referenced.

Define $\boxed{\text{Coordinator } C}$

$$\vec{K_t} = svd(TEncoder(T))$$

$$C(\vec{K_t}) = \Phi(t)$$

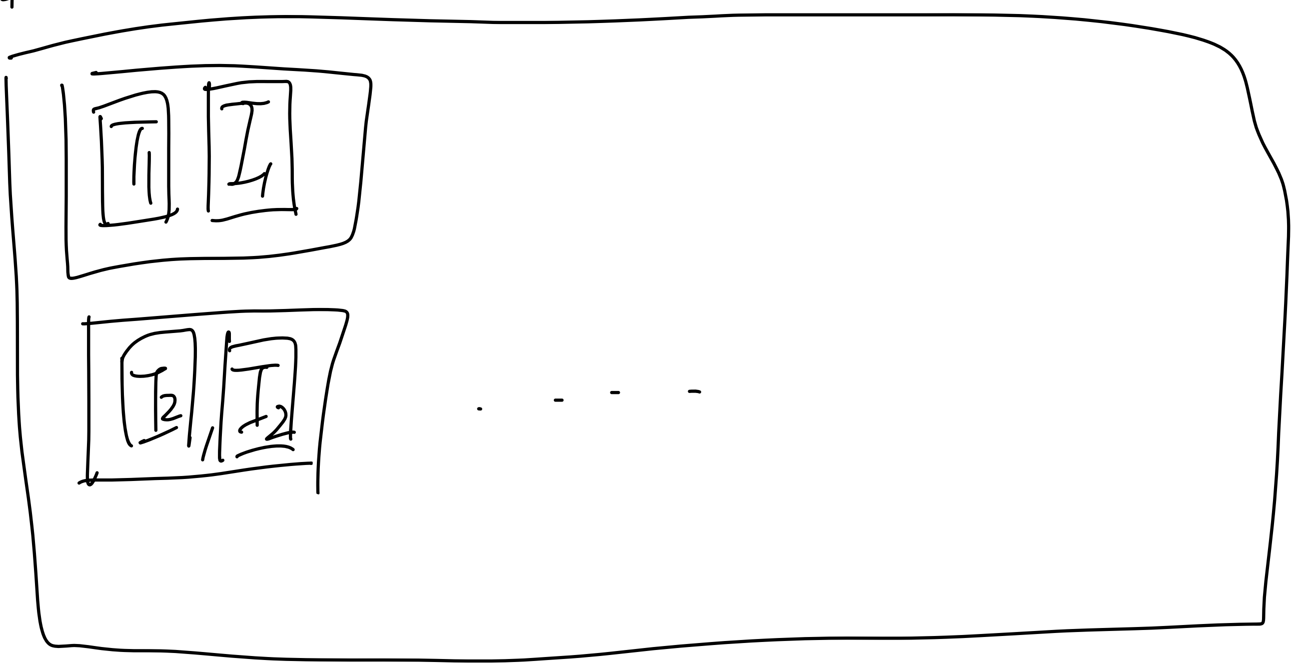
$$\Rightarrow P(t, \Phi(t))$$

Influence Prob

Determine a set of data with cons.

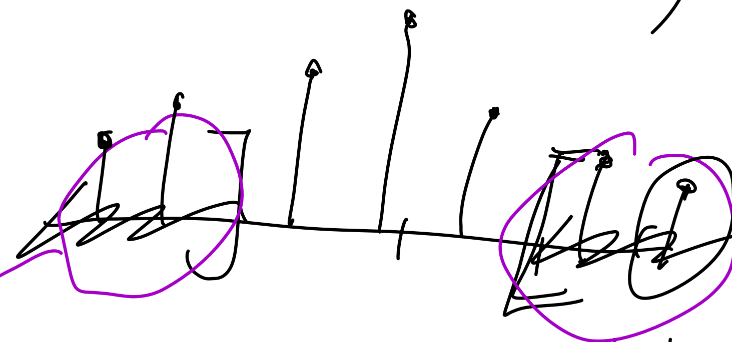
to learn weights to train loss function

Data =



$囲 = \mathrm{TEncoder}(\mathrm{Text})$

$(I_1, I_2, \ldots) \xrightarrow{\text{Normalized}} (i_1, i_2, i_3, \ldots)$

$囲 \longleftrightarrow$ 

$\mathrm{svd}(囲) = (k_1, k_2, k_3, \ldots)$

$\max \mathrm{svd}(囲)$
s.t. $(i_1, i_2, i_3, \ldots)$

$\min \mathrm{svd}(囲)$
s.t. $(i_1', i_2', \ldots)$

$\max \operatorname{svd}(T_N)$
s.t. i_N

?

$\{w_{N,i}\}_i$ $\quad$ $\max \operatorname{svd}(T_{N-1})$
s.t. i_{N-1}

$\{W_{N-1,i}\}_i$

$\mathrm{TEncode}(T) = \boxed{\;}$

$\vec{W}_N \quad \vec{W}_{N-1} \quad \vec{W}_{N-2} \quad \cdots \quad \vec{W}_1$

$$\begin{bmatrix} W_{N,1} \\ \vdots \\ \vdots \end{bmatrix} \leftrightarrow \begin{bmatrix} \\ \\ \\ \end{bmatrix} \leftrightarrow \begin{bmatrix} \\ \\ \\ \end{bmatrix} \leftrightarrow \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

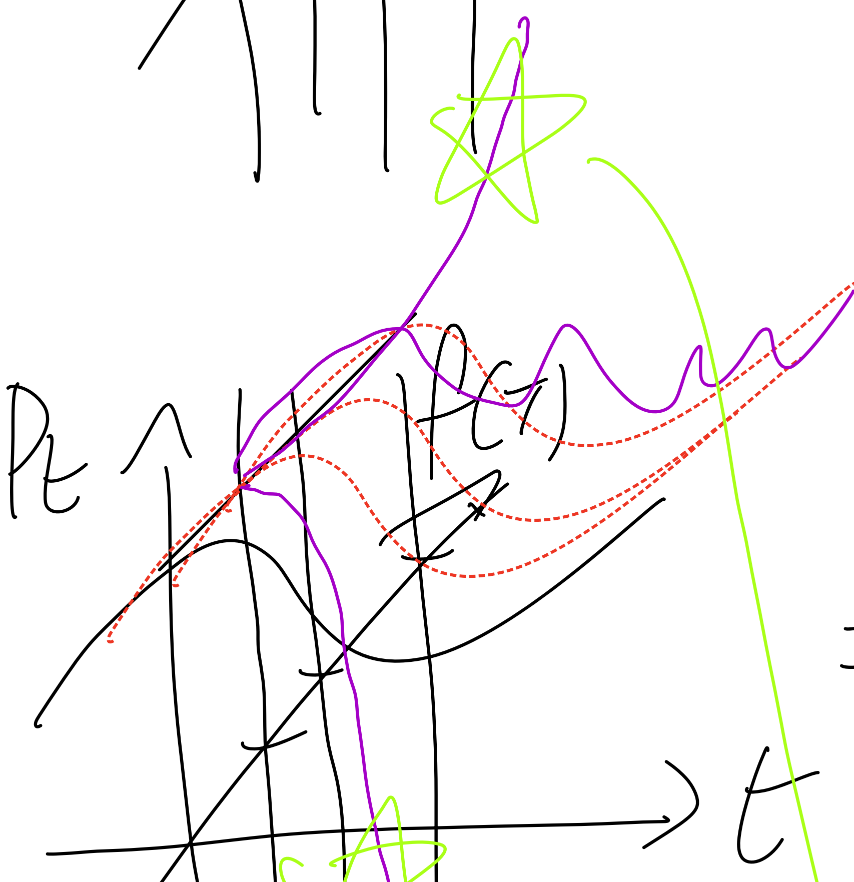
Interpolation

$\boxed{T} \longrightarrow \boxed{f(T)}$ Reaction Mechanism

correspondence build

I

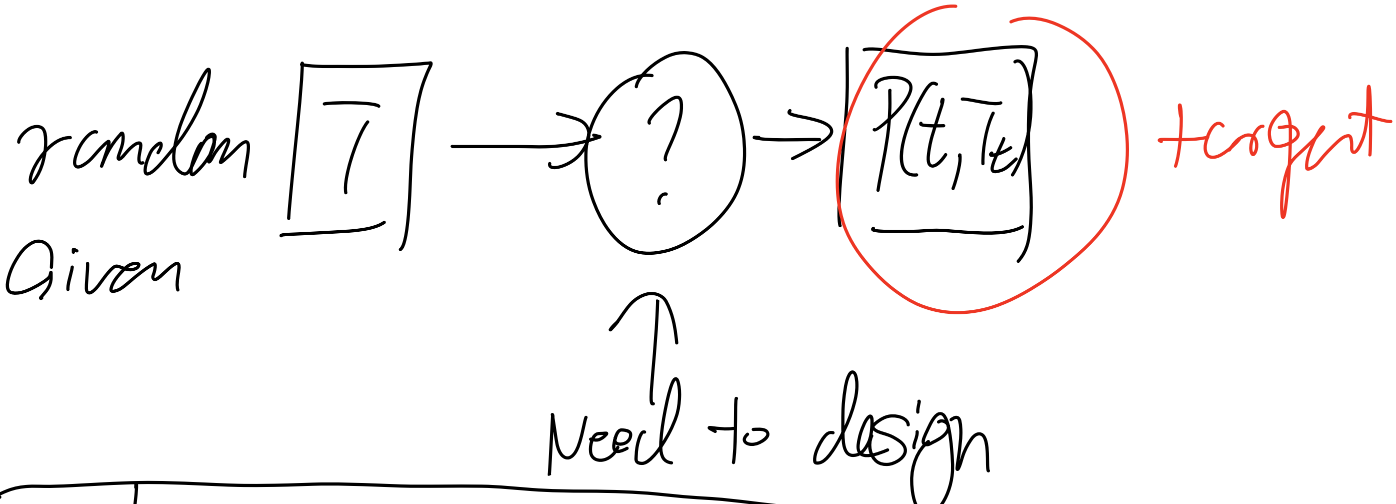
Penalize



$\Rightarrow \{\{w_i\}_i^k\}$

$f(T, t) \in (\mathbb{R}^{384}, \mathcal{N}^t)$

Revisit Inference Problem

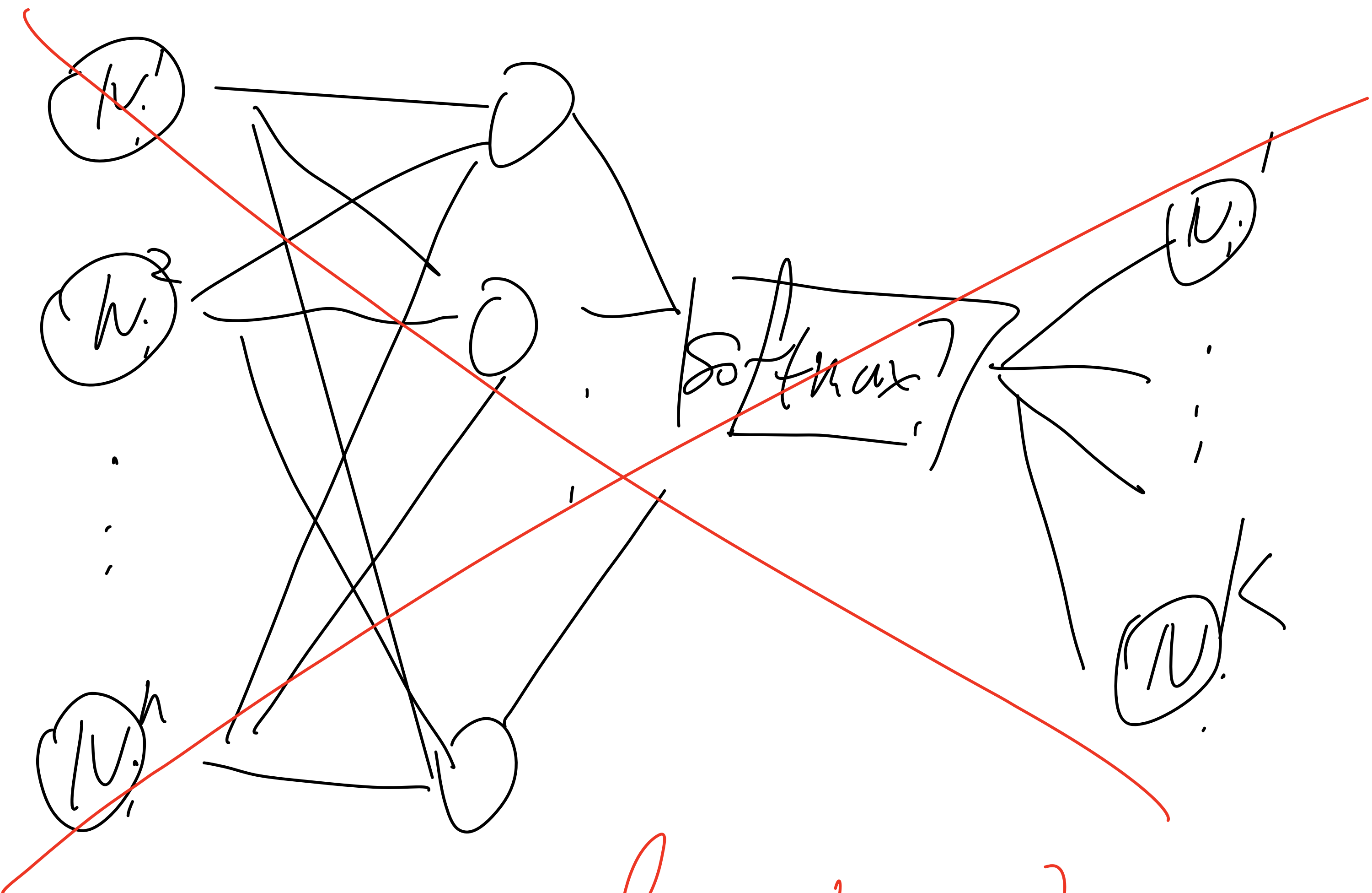


Remark:

PINN of Inverse $\boxed{\text{Backward Equation}}$

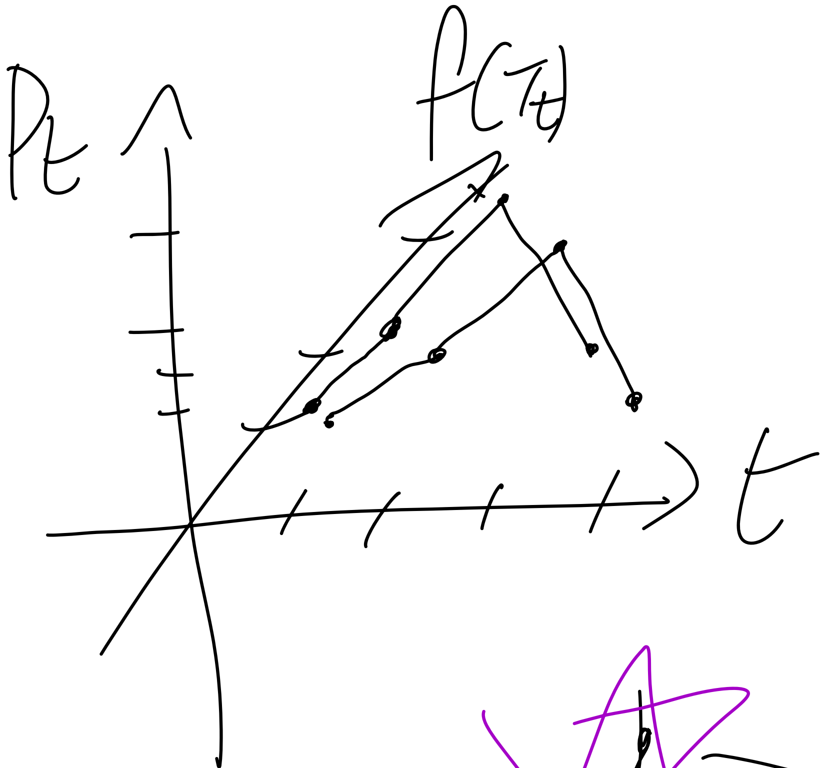
$$\boxed{T} \rightarrow \theta \rightarrow \boxed{P_\theta(t)} \leftarrow \, ? \, \leftarrow \boxed{p(t)}$$

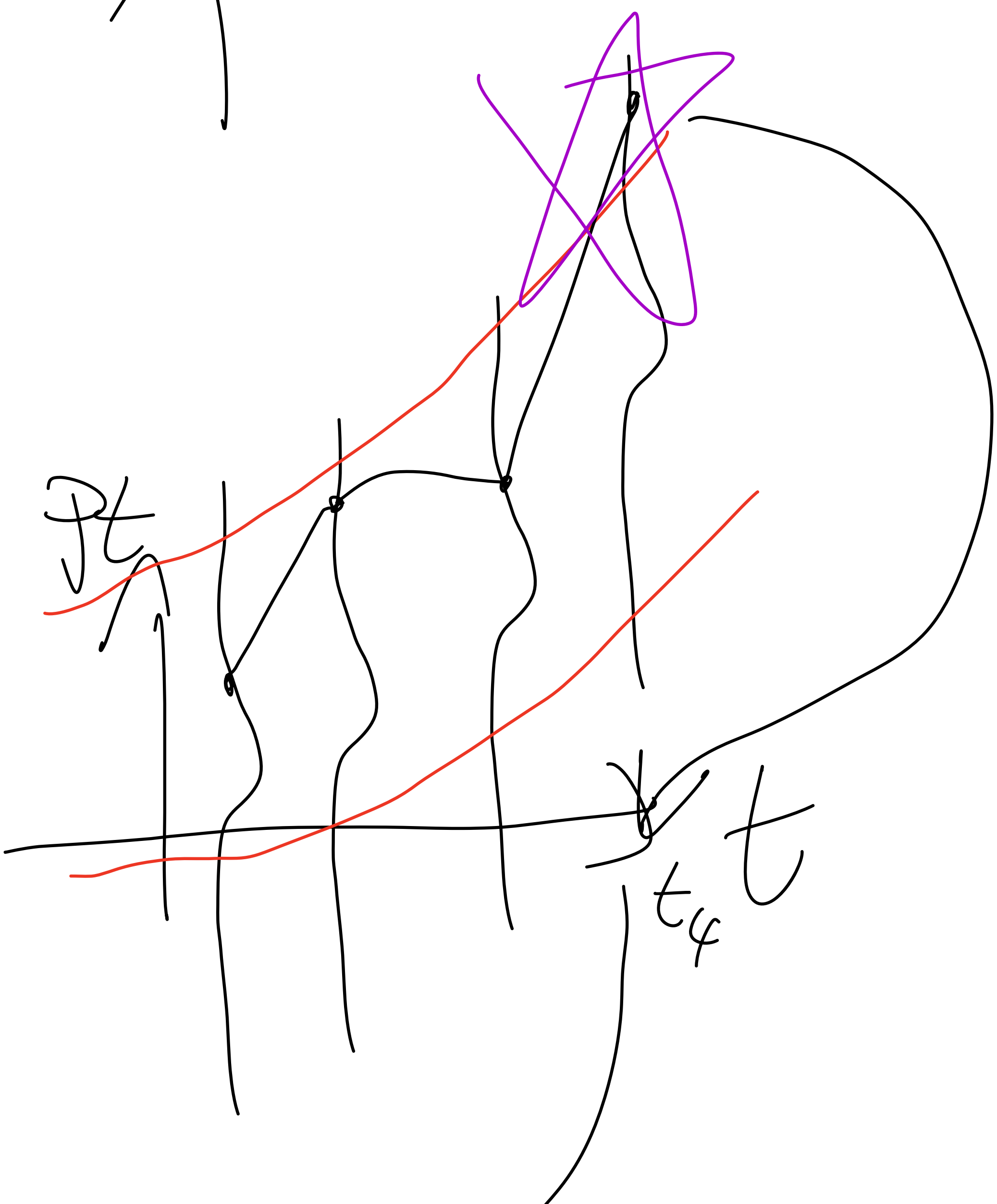
Given → target ← Inferral ← Given



Solution: unified NN?

(II) Detection method





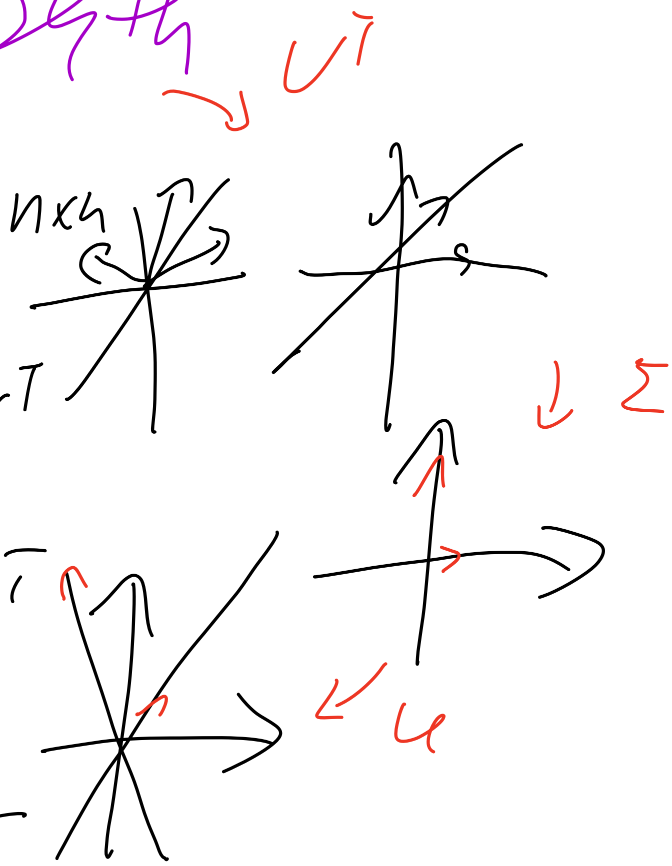
$\boxed{T_{[T_3, T_4]}}$

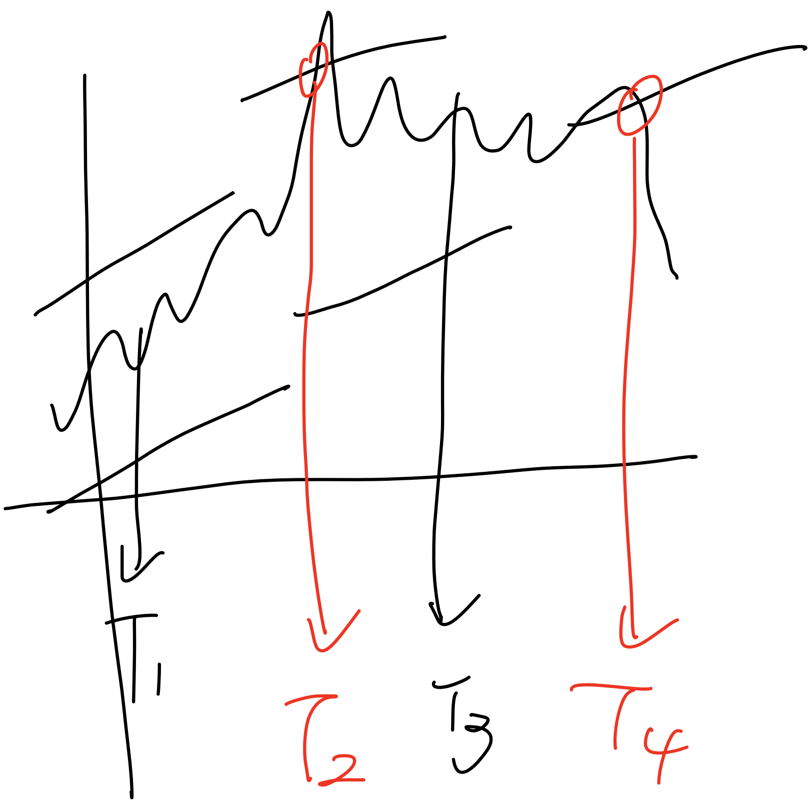
SVD Explain July 24th

$$\overset{m\times n}{A} = \overset{m\times m}{U}\,\Sigma\,\overset{n\times n}{V^T}$$

$$S_C = A^T A = V\Sigma^T U^T U \Sigma V^T$$

$$S_R = A A^T = U \Sigma \underline{V^T V} \Sigma^T U^T$$



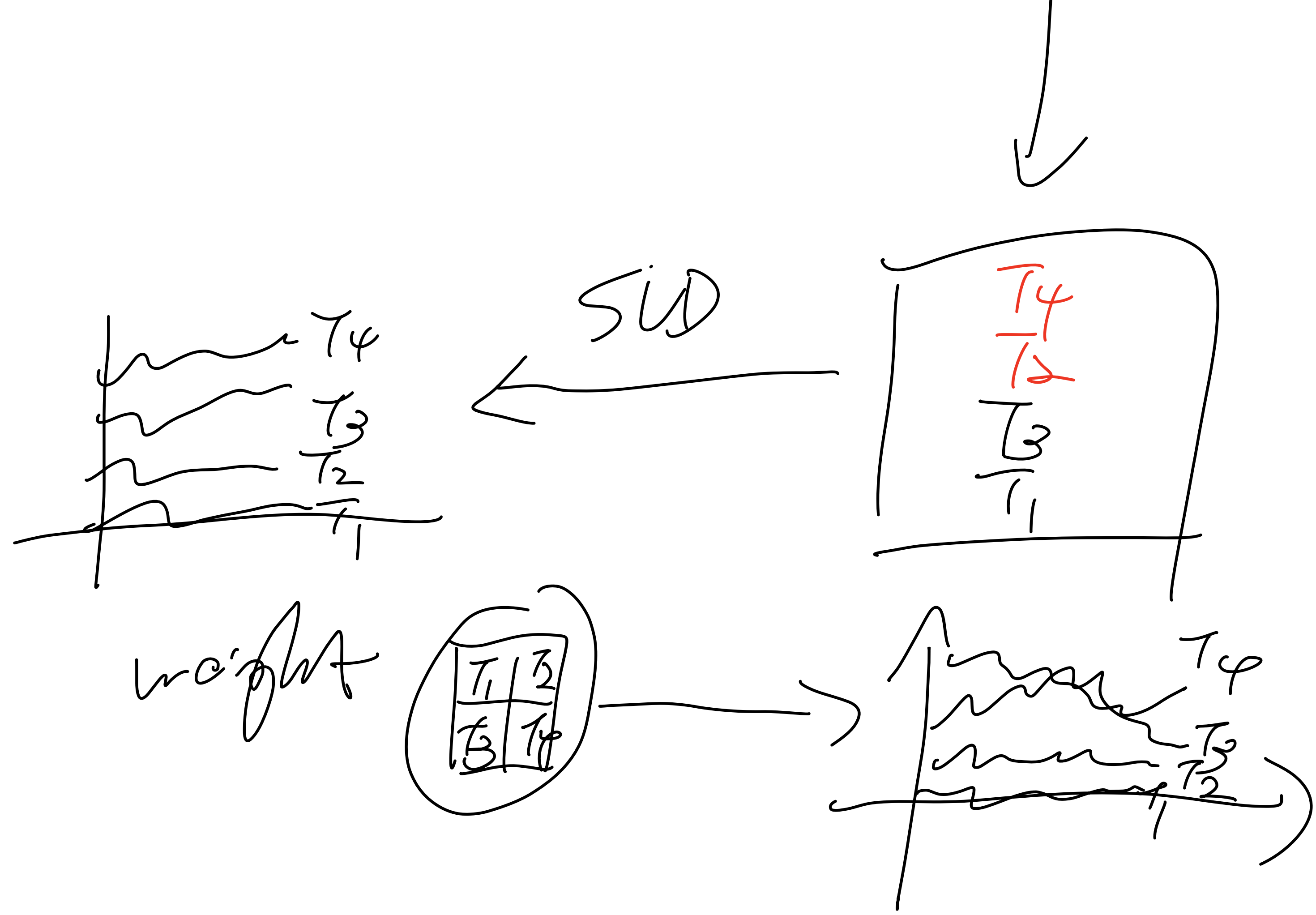


T_1

T_2

T_3

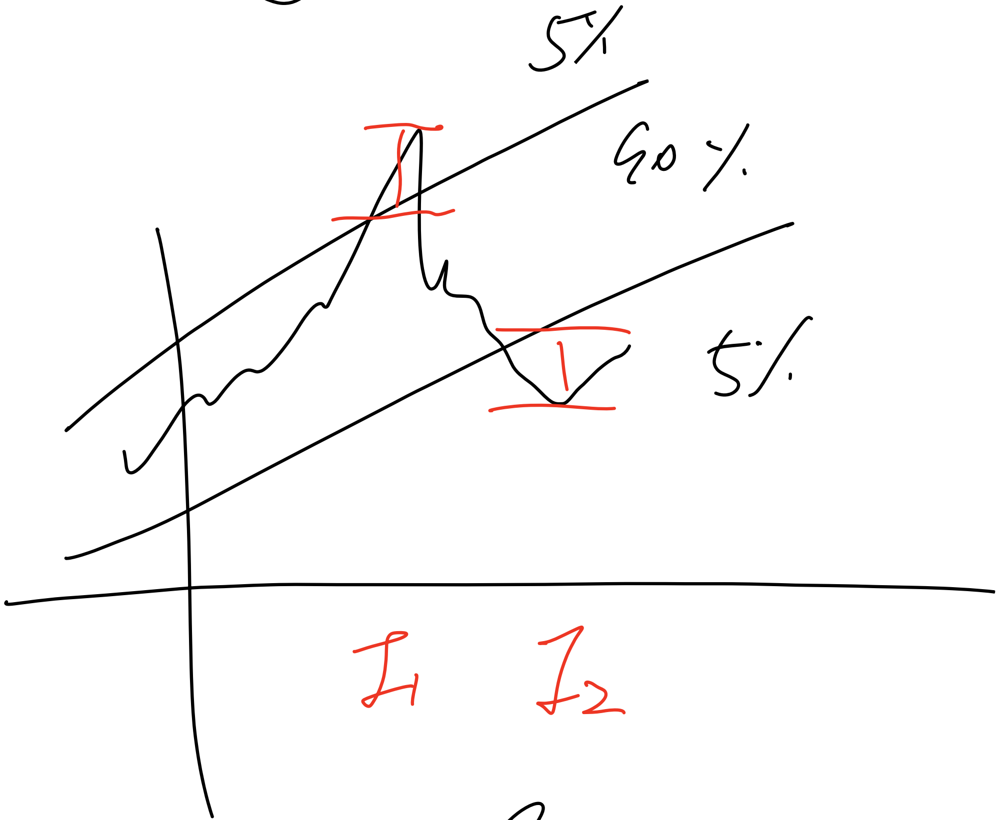
T_4



① weights to Topic

② Reorder

Matching



I = counts of best × out value

$f(T) = \sum_{std}(\text{Evalor}(T))$

Out ← 10%

Out ← $f(T)$

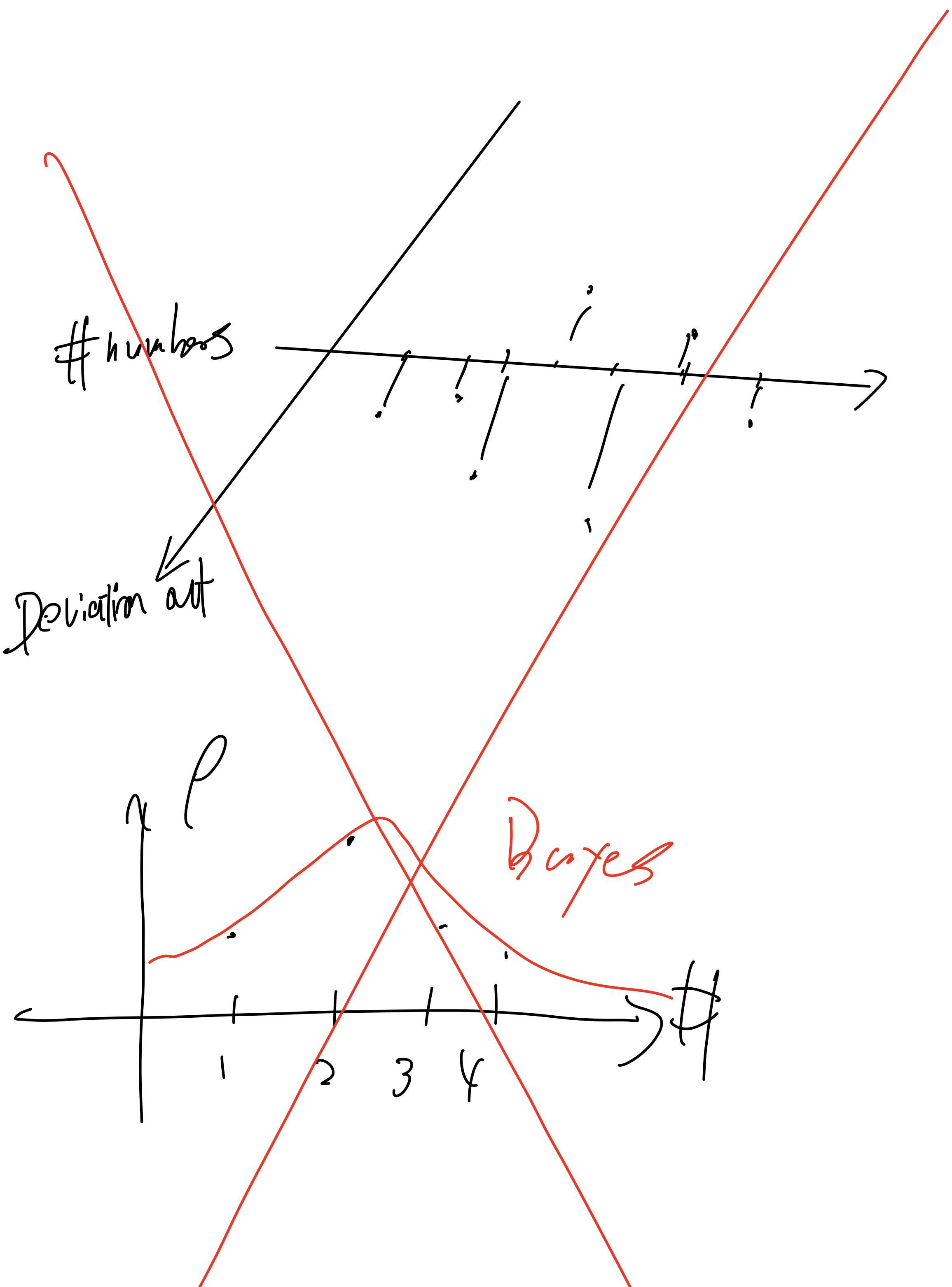
If no $f(T)$,

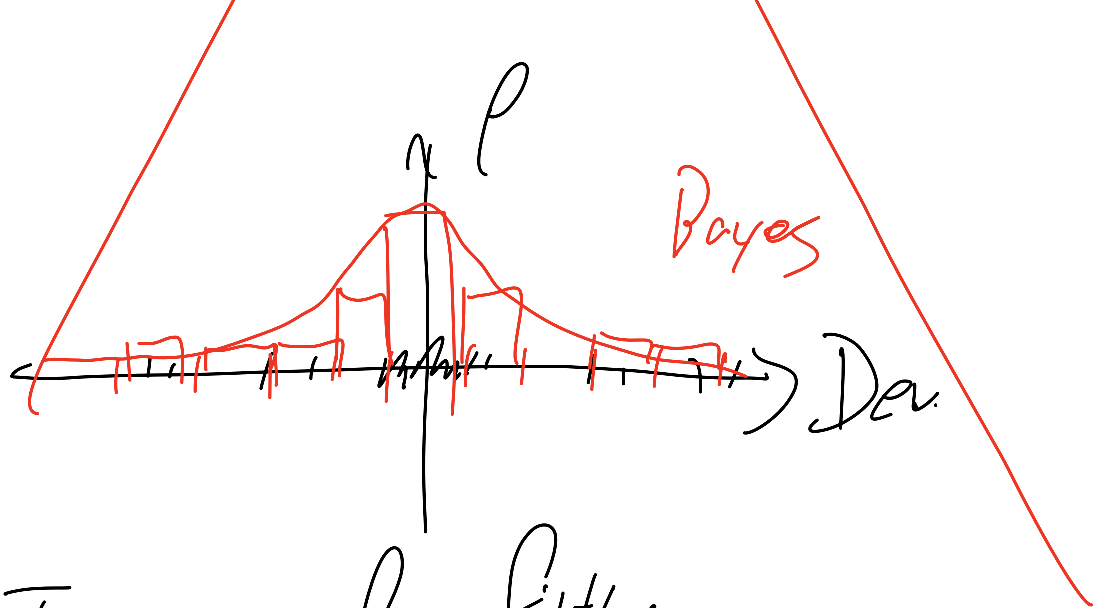
LLN gives 90 in 10 out on avg.

so, if we sampled in history data

80 in 20 out

we have 10 out counted towards $f(T)$





Training for fitting

$$f(T) \longrightarrow P^{out}(t, f(T_t))$$

$10\% \nearrow$

Important Scores $\{I_i\}$

$$dP_t = \left(\sum_{i=1}^{N} \alpha_i I_i\right) b(t, P_t)\,dt + \left(\sum_{i=1}^{N} \beta_i I_i\right) \sigma(t, P_t)\,dW_t$$

Let $\tilde{b}(f(T_t), t, P_t) = \left(\sum_{i=1}^{N} \alpha_i I_i\right) b$

$$\tilde{\sigma}(f(T_t), t, P_t) = \left(\sum_{i=1}^{N} \beta_i I_i\right) \sigma$$

Params

$\{\alpha_i ; \beta_i\}$ train

Then

$$dP_t = \tilde{b}\, dt + \tilde{\sigma}\, dW_t$$

Compact Interval

Stone-Weierstrass

guarantees

Approximation

GBM

$$dP_t = \tilde{\mu} S_t\, dt + \tilde{\sigma} S_t\, dW_t$$

Then, Fokker-Planck is

$$\begin{cases} \partial_t \rho = \mathcal{L}^{*} \rho & (t > 0) \\ \rho(t, 0) = \rho_0(x) \end{cases}$$

$$\mathcal{L}^{*} g := -\nabla(\tilde{\mu} s\, g(s)) + \nabla^2 : (a(s,t)\, g(s))$$

with $\nabla^2 : (ag) = \sum_{ij} \partial_{ij}(a_{ij} g) = \mathrm{tr}(H_{ag})$

$$a = \frac{1}{2} \sigma \sigma^{T} = \frac{1}{2} \tilde{\sigma}^2 S^2$$

$$\partial_t \rho = \mathcal{L}^{*} \rho = -\partial_x(\tilde{\mu} s\, g(s)) + \partial_x^2 \left(\frac{1}{2} \tilde{\sigma}^2 S^2 g(s)\right)$$

$$\rho = \int p(x,t|y,0)\,\rho_0(y)\,dy$$

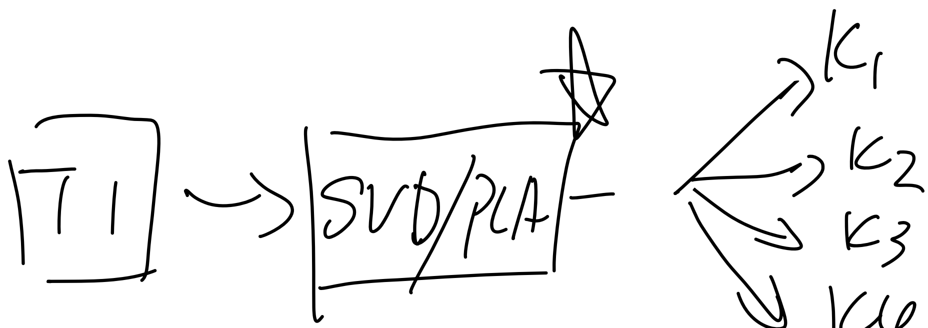
Transition Density

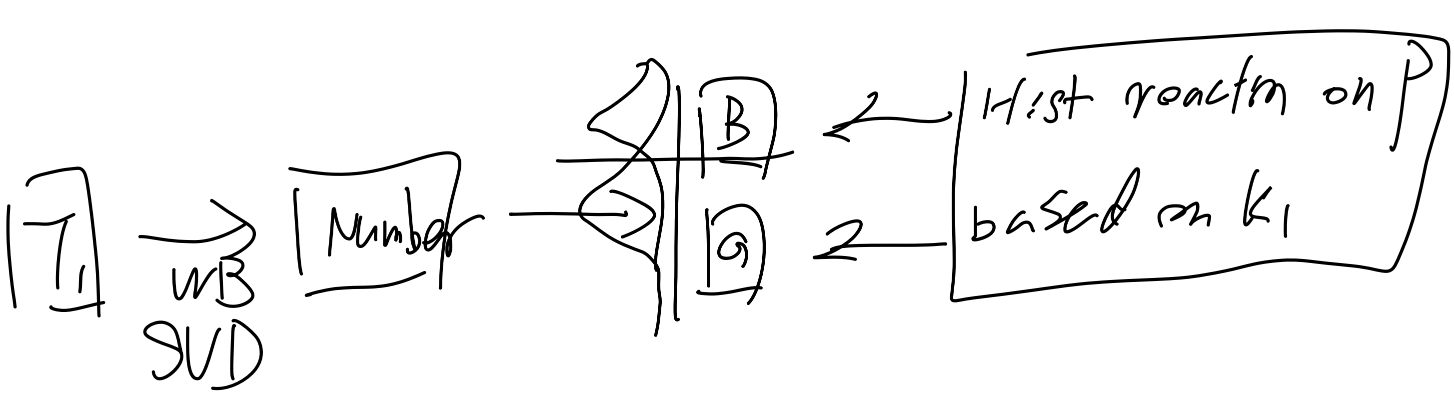
$$\begin{cases} \partial_t p = \mathcal{L}_s^* p, & p(x,t|y,t_0) \\ p(x,s|y,s) = \delta(x-y) \end{cases}$$

transition density

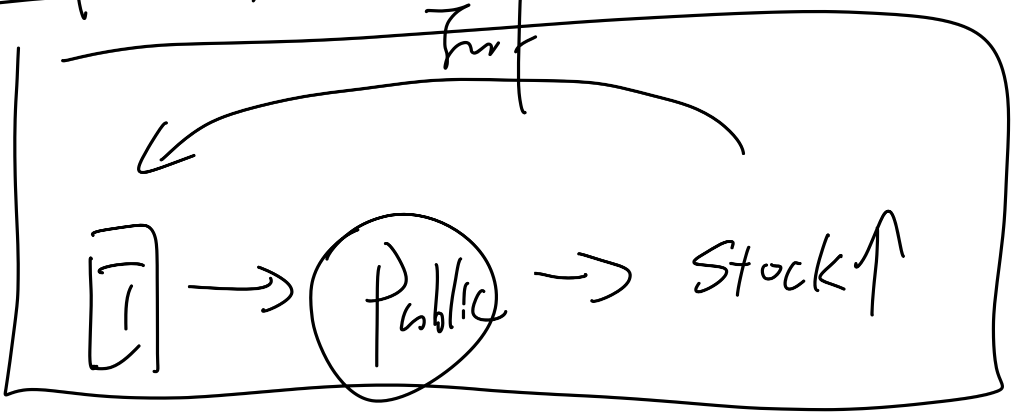
loss function

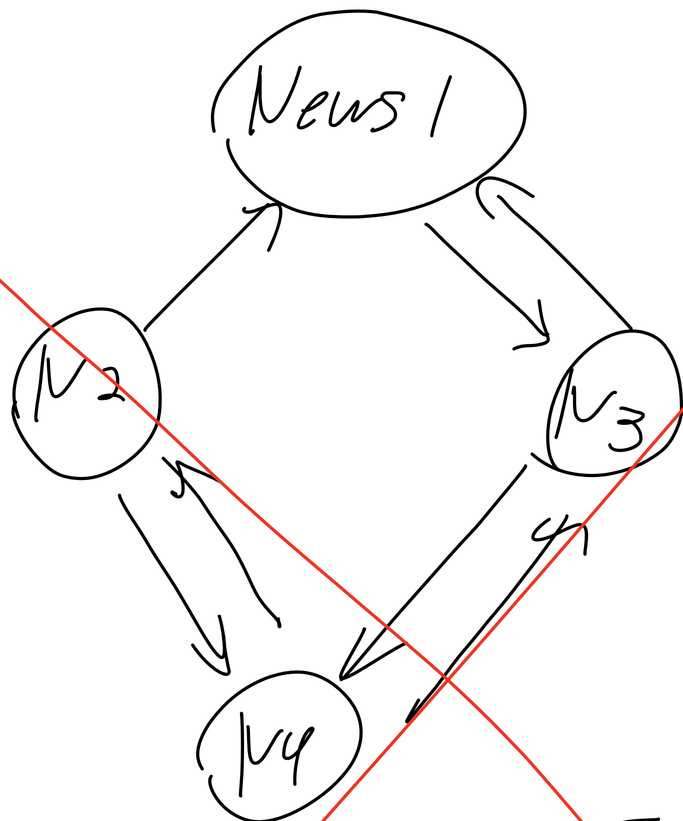
$$\|\partial_t p - \mathcal{L}_s^* p\|$$


T1
SVD/PCA
K1
K2
K3
K4


T1
WB
SVD
Number
B
G
First reaction on P
based on K1

Influence





$$\Rightarrow \begin{array}{cc} & \begin{matrix}1 & 2 & 3 & 4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix} & & & \\ & \text{Matrix} & & \\ & & & \end{bmatrix} \end{array} \Rightarrow$$

Eigenval $\lambda = 1$

$$V = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}$$

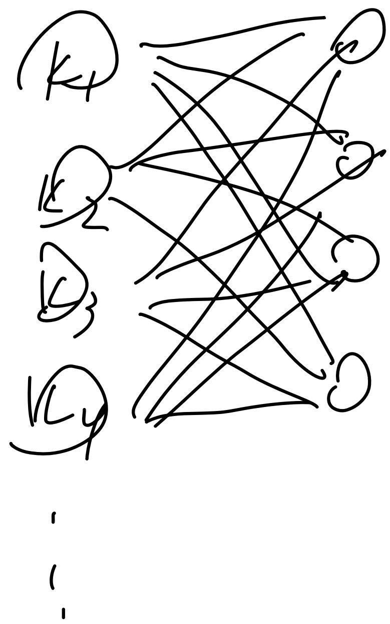
$$r_1 = \frac{q_1}{\sum_i q_i}$$

Possible Factor : Fast , Huge

or use ML to see more general factors for agents

Like NN

random data



Goal : Effectiveness

$\boxed{T} \longleftrightarrow \boxed{PC}$