Symbolic networks

PDE Net 2.0, постановка задачи

Предположим, мы хотим описать системой уравнений в частных производных набор данных, полученный экспериментально.

В биологии к таким задачам относится, например, задача фармакокинетики.

Мы полагаем что наша система будет иметь вид:

$$U_t = F(U, \nabla U, \nabla^2 U, \ldots), \quad x \in \Omega \subset \mathbb{R}^2, \quad t \in [0, T].$$

У нас есть экспериментальные данные со значением функции U в некотором объеме (или плоскости) на заданном интервале времени.

И нам необходимо знать какую форму принимают уравнения нашей системы, т.е. определить функцию F

PDE Net 2.0

PDE-NET 2.0: LEARNING PDES FROM DATA WITH A NUMERIC-SYMBOLIC HYBRID DEEP NETWORK

Zichao Long
School of Mathematical Sciences
Peking University
zlong@pku.edu.cn

Yiping Lu
School of Mathematical Sciences
Peking University
luyiping9712@pku.edu.cn

Bin Dong
Beijing International Center for Mathematical Research
and Center for Data Science
Peking University
dongbin@math.pku.edu.cn

PDE Net 2.0

$$U_t = F(U, \nabla U, \nabla^2 U, \ldots), \quad x \in \Omega \subset \mathbb{R}^2, \quad t \in [0, T].$$

$$\tilde{U}(t + \delta t, \cdot) \approx \tilde{U}(t, \cdot) + \delta t \cdot SymNet_m^k(D_{00}\tilde{U}, D_{01}\tilde{U}, D_{10}\tilde{U}, \cdots)$$

Здесь оператор D_{ij} это оператор свертки с фильтром q_{ij}, т.е. $D_{ij}\, u = q_{ij} \circledast u$. Операторы D_{10}, D_{01}, D_{11} и т. д. аппроксимируют дифференциальные операторы, т.е.

$$D_{ij} u \approx \frac{\partial^{i+j} u}{\partial^i x \partial^j y}$$

Оператор D_{00} - оператор усреднения.

PDE Net 2.0

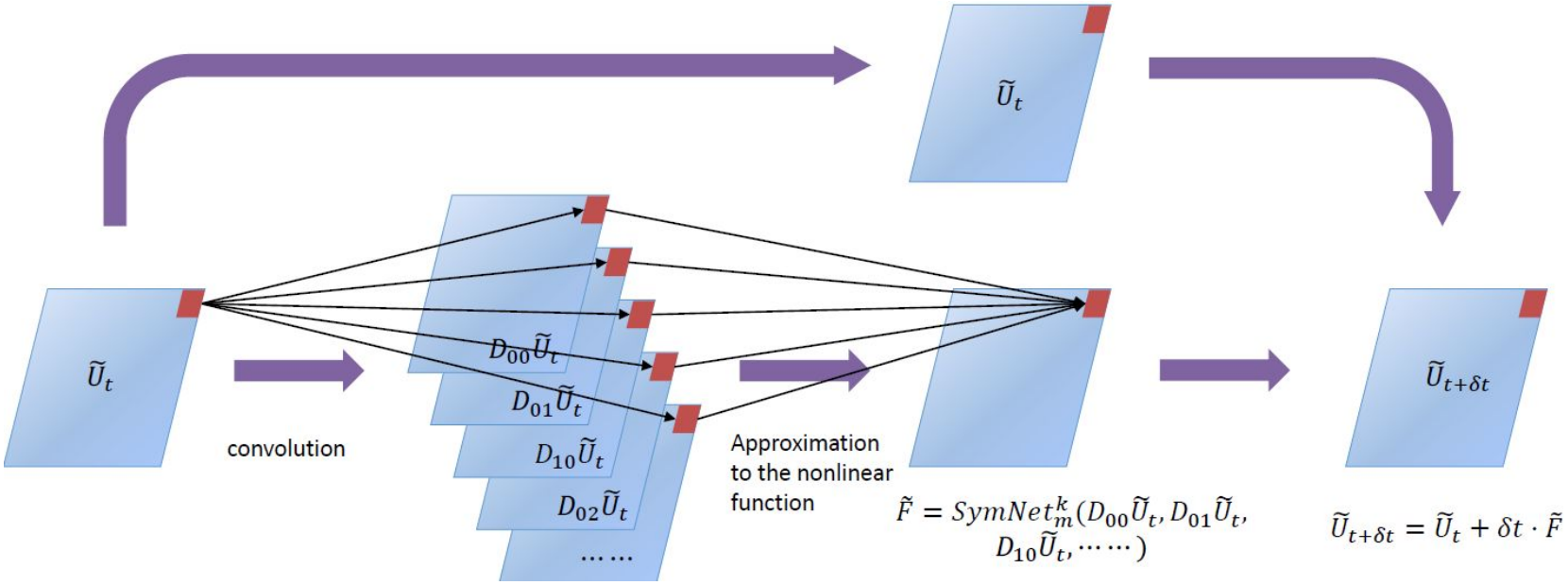


Схема расчета

PDE Net 2.0

Для фильтра q размером NxN определим матрицу моментов как

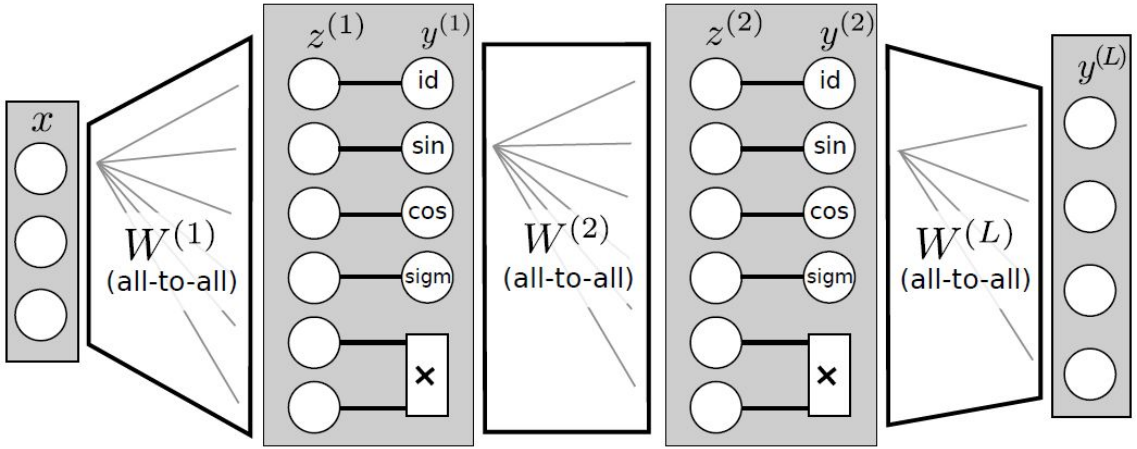
$$M(q) = (m_{i,j})_{N\times N}, \quad m_{i,j} = \frac{1}{i!j!} \sum_{k_1,k_2=-\frac{N-1}{2}}^{\frac{N-1}{2}} k_1^i k_2^j q[k_1,k_2], i,j = 0,1,\ldots,N-1$$

Для любой гладкой функции f : $R^2 \to R$, мы применяем свертку к кусочной интерполяции f относительно фильтра q. Используя разложение Тейлора, можно легко получить следующую формулу

$$\begin{aligned}\sum_{k_1,k_2=-\frac{N-1}{2}}^{\frac{N-1}{2}} q[k_1,k_2] f(x+k_1\delta x, y+k_2\delta y) &= \sum_{k_1,k_2=-\frac{N-1}{2}}^{\frac{N-1}{2}} q[k_1,k_2] \sum_{i,j=0}^{N-1} \frac{\partial^{i+j} f}{\partial^i x \partial^j y}\bigg|_{(x,y)} \frac{k_1^i k_2^j}{i!j!} \delta x^i \delta y^j + o(|\delta x|^{N-1} + |\delta y|^{N-1}) \\ &= \sum_{i,j=0}^{N-1} m_{i,j} \delta x^i \delta y^j \cdot \frac{\partial^{i+j} f}{\partial^i x \partial^j y}\bigg|_{(x,y)} + o(|\delta x|^{N-1} + |\delta y|^{N-1}). \end{aligned} \quad (2)$$

Т.е. фильтр q можно спроектировать для аппроксимации любого дифференциального оператора с заданным порядком точности, наложив ограничения на M(q).

PDE Net 2.0



$$f_i(z_i) := \begin{cases} z_i \\ \sin(z_i) \\ \cos(z_i) \\ \mathrm{sigm}(z_i) \end{cases}$$

$$g_j(z_{u+2j-1}, z_{u+2j}) := z_{u+2j-1} \cdot z_{u+2j}$$

i = 1,....,u; j = 1,....,v

Архитектура Equation Learner (EQL) для L = 3, u = 4, v = 1.
Нелинейный этап имеет u + v выходов и u + 2v входов.

$$z^{(l)} = W^{(l)} y^{(l-1)} + b^{(l)} \quad y^{(l)} := \left(f_1(z_1^{(l)}), f_2(z_2^{(l)}), \ldots, f_u(z_u^{(l)}),\ g_1(z_{u+1}^{(l)}, z_{u+2}^{(l)}), \ldots, g_v(z_{u+2v-1}^{(l)}, z_{u+2v}^{(l)}) \right)$$

PDE Net 2.0

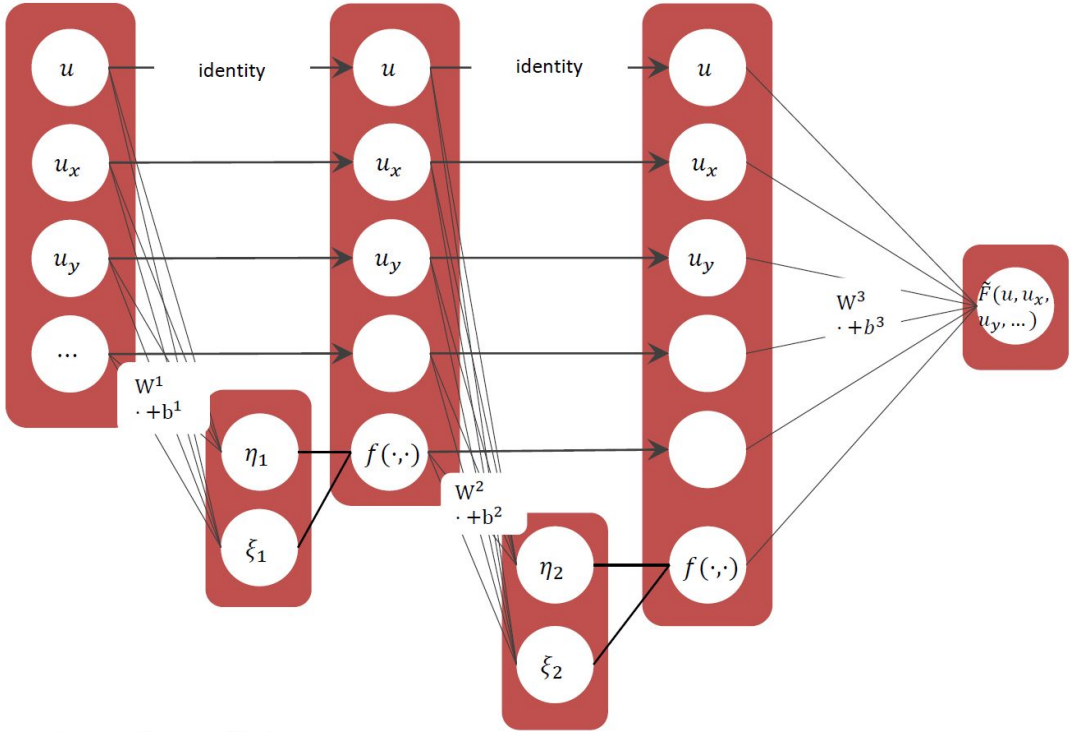
$W^1 =$

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1
1	0	0	0	0	0
0	1	0	0	0	0

$W^2 =$

1	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	1
0	0	1	0	0	0	0
0	0	0	1	0	0	0

$W^3 =$ 0 0 0 0 0 0 -1 -1



$$SymNet_6^2(u, u_x, u_y, v, v_x, v_y) = -uu_x - vu_y$$

PDE Net 2.0

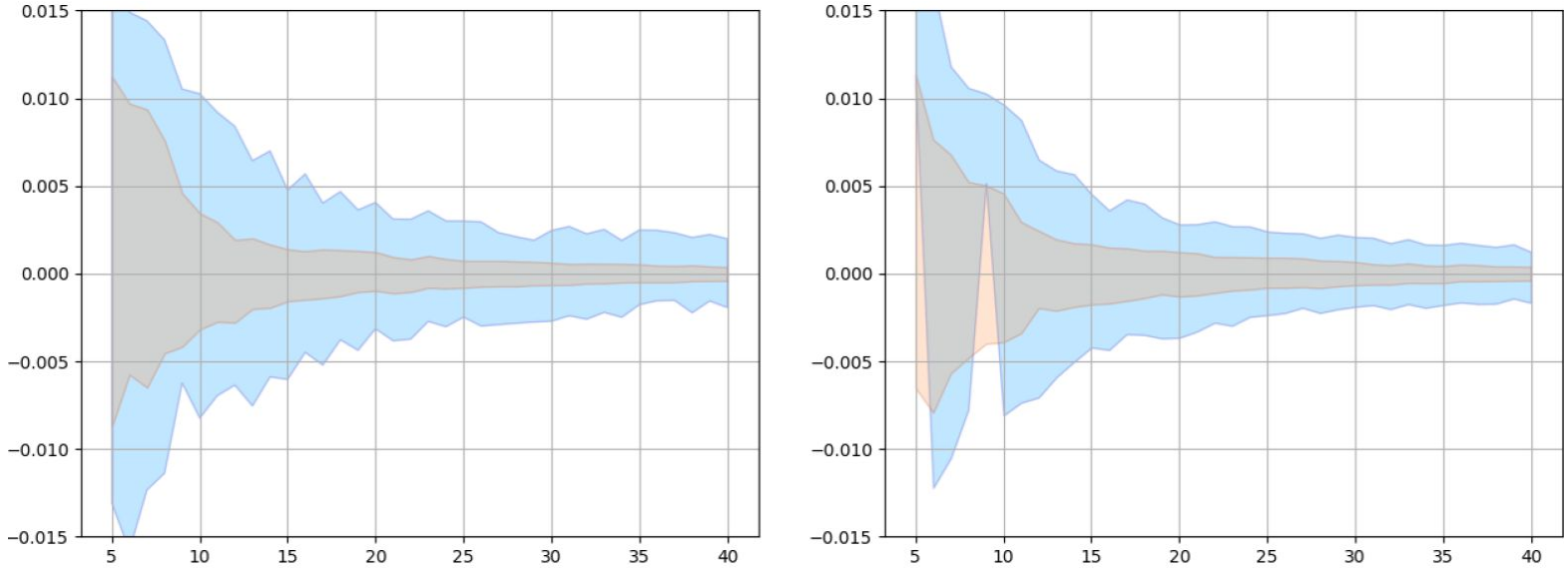
Тестирование PDE Net проведено на двумерном уравнении Бюргера с периодическим граничным условием Ω = [0; 2π] x [0; 2π],

$$\begin{cases} \frac{\partial U}{\partial t} & = -U \cdot \nabla U + \nu \Delta U, U = (u, v)^\top \\ U|_{t=0} & = U_0(x, y), \end{cases}$$

Данные обучения сгенерированы с добавлением шума в начальные условия. Кроме того шум добавлен и полученным значениям U = (u; v)^T.
Порядок частных производных ограничен 2. Авторы использовали два SymNet$^5_{12}$ для покомпонентной аппроксимации правой нелинейной функции и получили следующий результат:

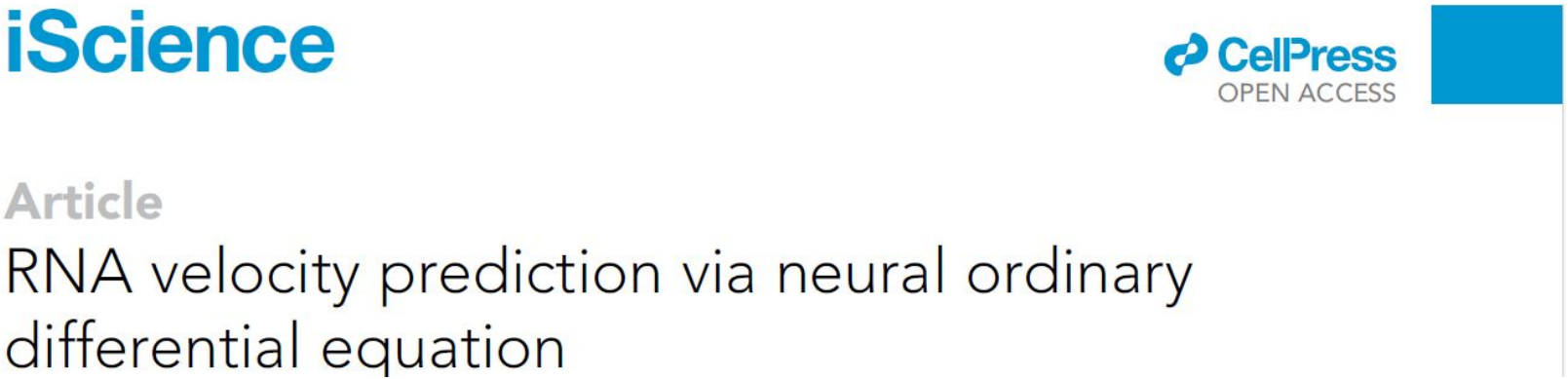
Correct PDE	$u_t = -uu_x - vu_y + 0.05(u_{xx} + u_{yy})$ $v_t = -uv_x - vv_y + 0.05(v_{xx} + v_{yy})$
Frozen-PDE-Net 2.0	$u_t = -0.906uu_x - 0.901vu_y + 0.033u_{xx} + 0.037u_{yy}$ $v_t = -0.907vv_y - 0.902uv_x + 0.039v_{xx} + 0.032v_{yy}$
PDE-Net 2.0	$u_t = -0.979uu_x - 0.973u_yv + 0.052u_{xx} + 0.051u_{yy}$ $v_t = -0.973uv_x - 0.977vv_y + 0.053v_{xx} + 0.051v_{yy}$

PDE Net 2.0



Коэффициенты остаточных членов для u-компонента (слева) и v-компонента (справа) с ограничением разреженности в SymNet (оранжевый) и без ограничения разреженности (синий). Полосы для обоих случаев рассчитаны на основе 20 независимых тренировок.

RNA динамика



RNA динамика

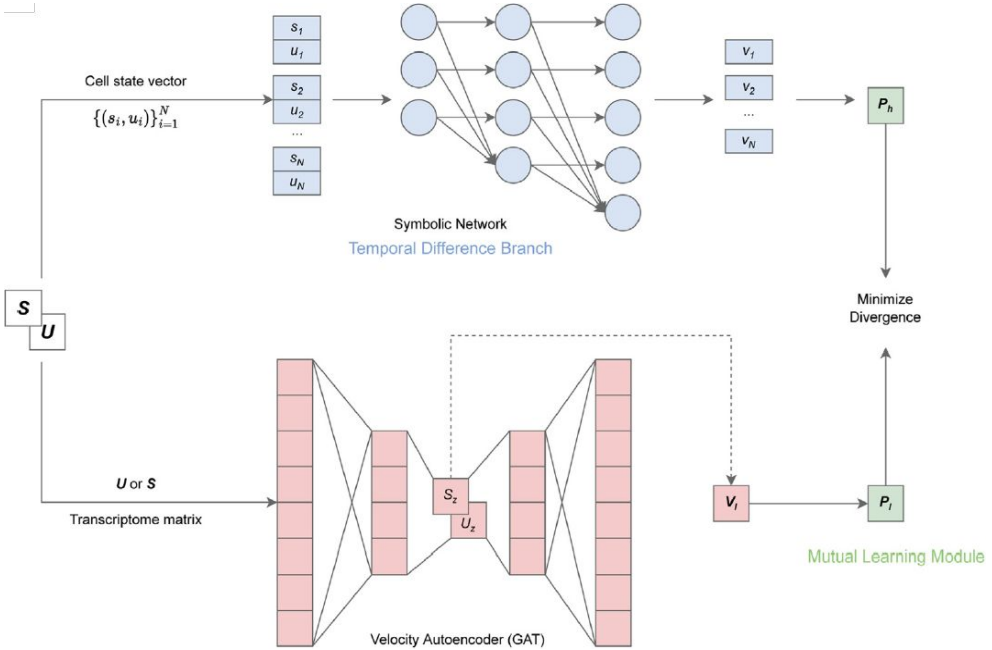
Начальный подход к моделированию динамики RNA (для каждого гена отдельно):

$$\frac{du}{dt} = \alpha(t) - \beta u(t)$$

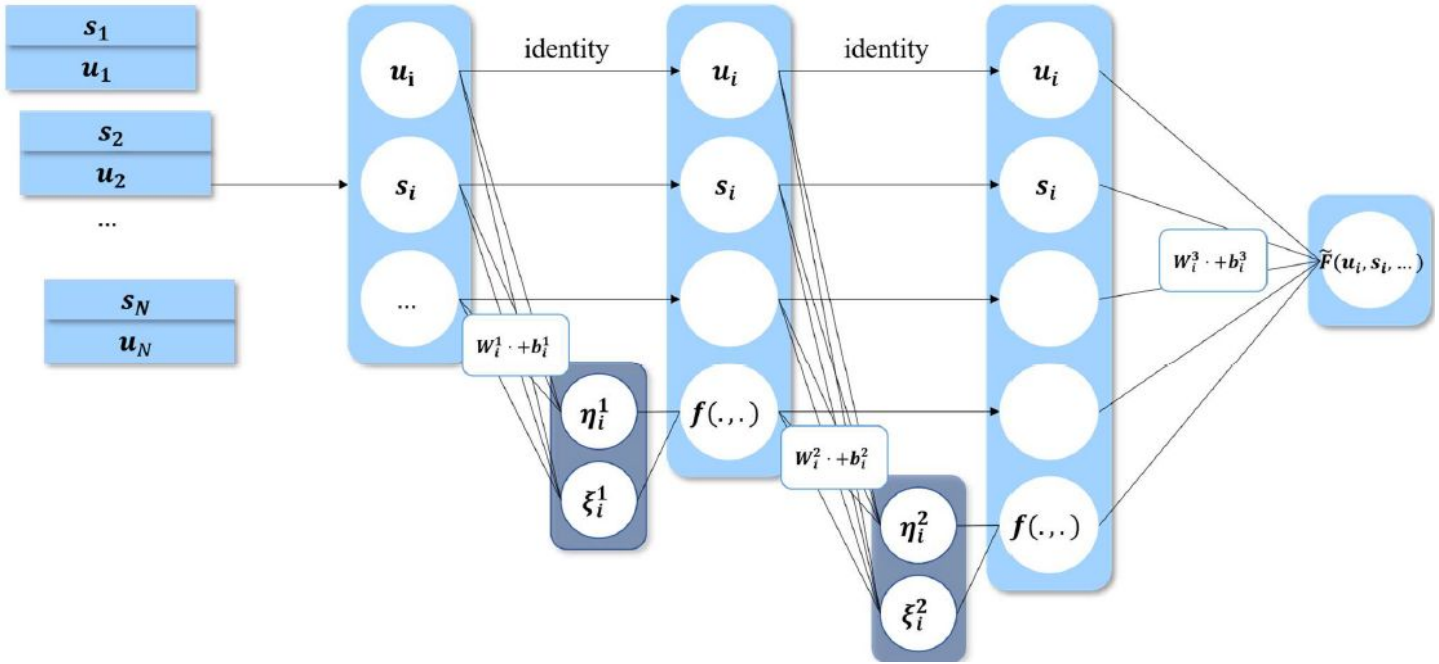
$$\frac{ds}{dt} = \beta u(t) - \gamma s(t)$$

где u(t), s(t) представляют собой нормированные концентрации несплайсированной и сплайсированной мРНК. α(t) - скорость транскрипции гена, β - скорость сплайсинга, ɣ - скорость деградации.

Подход, предложенный авторами:



RNA динамика



Нейросети и символические обучающиеся системы

Восприятие, представленное нейронными системами, и познание, представленное символизмом или символическими системами, являются двумя фундаментальными парадигмами в области искусственного интеллекта (ИИ), каждая из которых преобладала на протяжении нескольких десятилетий.

Systems	Processing Methods	Knowledge representation	Primary algorithms	Advantages	Disadvantages
Symbolic systems	Deductive reasoning	Logical representation	Logical deduction	● Strong generalization ability ● Good interpretability ● Knowledge-driven	● Weak at handling unstructured data ● Weak robustness ● Slow reasoning
Neural systems (Sub-symbolic systems)	Inductive learning	Distributed representation	BP algorithms	● Strong at handling unstructured data ● Strong robustness ● Fast learning	● Weak generalizability (adaptability) ● Lack of interpretability ● Data-driven

Этот раздел лекции построен на публикациях:

1. A Survey on Neural-symbolic Learning Systems, Dongran Yu,Bo Yang,Dayou Liu,Hui Wang,Shirui Pan, 2023
2. Learn to explain efficiently via neural logic inductive learning, Yuan Yang & Le Song, Georgia Institute of Technology, 2020
3. Tunneling Neural Perception and Logic Reasoning through Abductive Learning, Wang-Zhou Daiy и др., Nanjing University, 2018

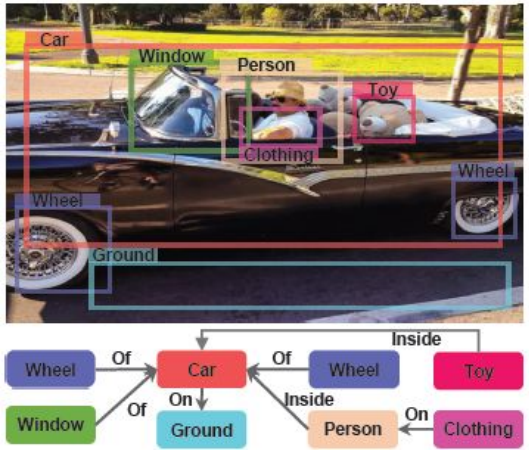
Логическое программирование

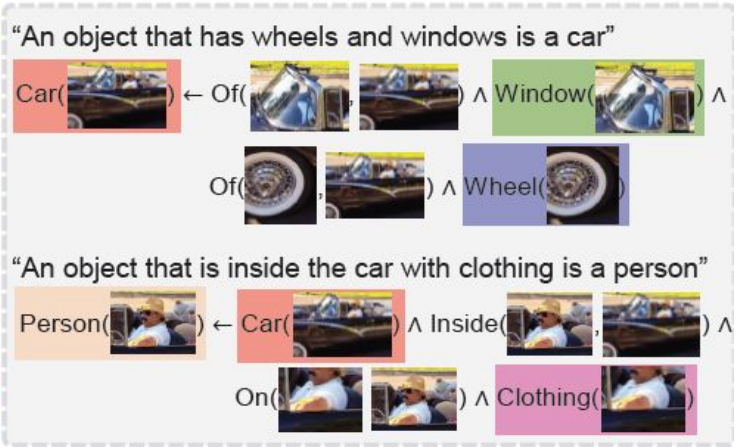
Логическая система первого порядка состоит из трех компонентов: сущности, предиката и правила.

Сущности это объекты x ϵ ***X***.

Предикаты это функции, которые сопоставляют сущности с 0 или 1, например. Person: x → {0,1}, x ϵ ***X***. Предикаты могут принимать несколько аргументов, например. Inside это предикат с двумя аргументами. Атом — это предикат, применяемый к логической переменной, например. Person (X) или Inside (X, X'). Логическая переменная X может быть инициализирована любым объектом x.

Правила логики первого порядка (FOL) представляют собой комбинацию атомов с использованием логических операций {Λ, V, ¬}.





Для множества предикатов P = {P1, …; PK}, мы определяем предикат Pk как FOL правило:

$$\forall X, X' \; \exists Pk\,(X, X') \leftarrow A\,(X, X', Y1; Y2,..)$$

,

$$\text{Person}\,(X) \leftarrow \text{Inside}\,(X; Y1) \wedge \text{Car}\,(Y1) \wedge \text{On}\,(Y2;X) \wedge \text{Clothing}\,(Y2)$$

KB - база знаний - состоит из набора фактов {⟨xi, Pk, xi' ⟩}, i = 1…N, k = 1…K, где xi, xi' ϵ X, Pk ϵ P

Learning for reasoning

Схема A

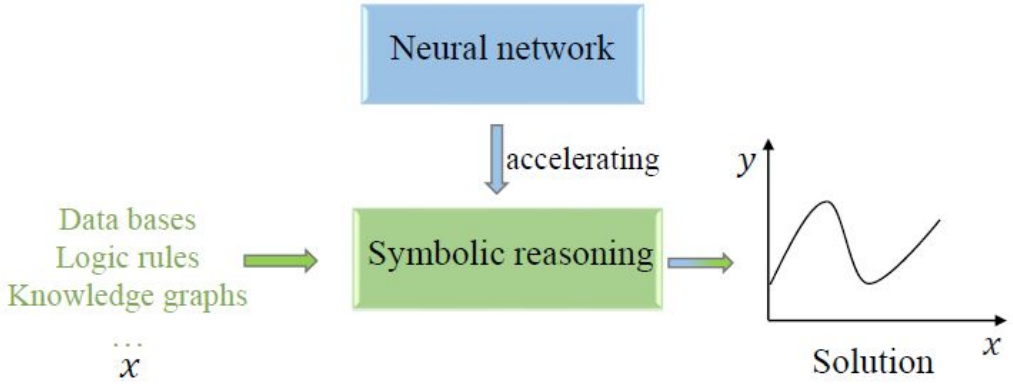
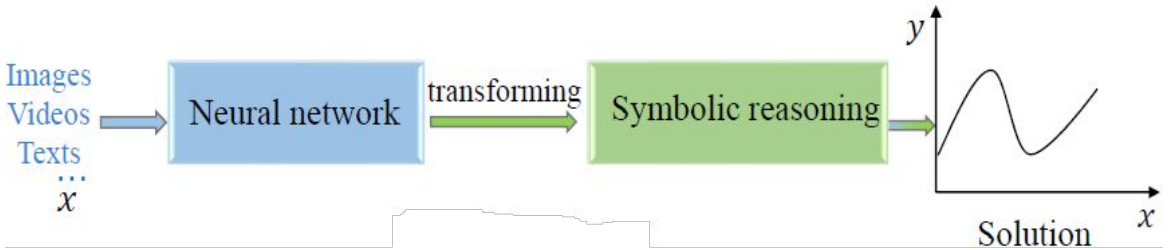
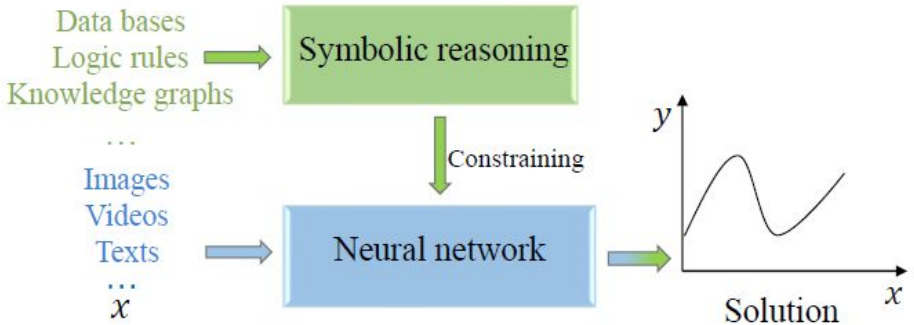


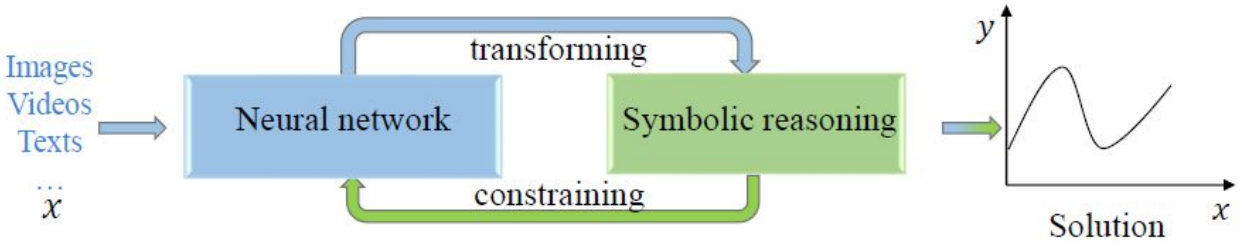
Схема B



Reasoning for learning



Learning-reasoning



Многошаговые рассуждения (MUTLI-HOP REASONING)

Задачу LP можно решать как задачу многошагового рассуждения на графе знаний. Задача на графе оперирует базой знаний, состоящей из набора предикатов P. Здесь объекты - узлы графа, предикаты - ребра. Предикат Pk можно представить в виде двоичной матрицы смежности Mk. Факт⟨xi, Pk, xj ⟩находится в базе данных тогда и только тогда, когда Mk [i, j] равна 1.

Запрос q =⟨x, P*, x'⟩сводится к поиску на графе пути $\mathbf{x} \xrightarrow{P^{(1)}} \ldots \xrightarrow{P^{(T)}} \mathbf{x}'$

Пусть Vx это one hot вектор объекта x с размерностью |X|. Тогда t-й шаг пути

$$\mathbf{v}^{(0)} = \mathbf{v}_{\mathbf{x}}, \qquad \mathbf{v}^{(t)} = \mathbf{M}^{(t)}\mathbf{v}^{(t-1)}$$

; где $M^{(t)}$ — матрица смежности предиката, используемого на t-м переходе. $v^{(t)}$ — это path features вектор, где j-й элемент $v^{(t)}j$ подсчитывает количество уникальных путей от x до xj

После T шагов рассуждения оценка запроса вычисляется как $\mathrm{score}(\mathbf{x}, \mathbf{x}') = \mathbf{v}_{\mathbf{x}'}^{\top} \prod_{t=1}^{T} \mathbf{M}^{(t)} \cdot \mathbf{v}_{\mathbf{x}}$

Для каждого q цель состоит в том, чтобы a) найти подходящее T и b) для каждого t = [1;2; :::; T], найти подходящее $M^{(t)}$ так, чтобы значение score было максимальным. Эти два дискретных выбора можно “смягчить”, обучая взвешенную сумму оценок всех возможных путей и взвешенную сумму матриц на каждом этапе.

Neural Logic Inductive Learning

Пусть

$$\kappa(\mathbf{s}_\psi, \mathbf{S}_\varphi) \equiv \sum_{t'=1}^{T} s_\psi^{(t')} \left(\prod_{t=1}^{t'} \sum_{k=1}^{K} s_{\varphi,k}^{(t)} \, \mathbf{M}_k \right)$$

это soft path selection функция, параметризованная a) path attention вектором: $\mathbf{s}_\psi = [s_\psi^{(1)}, ..., s_\psi^{(T)}]^\top$

который выбирает лучший путь длиной от 1 до T, и b) operator attention вектором: $\mathbf{S}_\varphi = [\mathbf{s}_\varphi^{(1)}, ..., \mathbf{s}_\varphi^{(T)}]^\top$

где $\mathbf{s}_\varphi^{(t)}$ выбирает $M^{(t)}$ на t-м шаге. Эти два вектора генерируются с помощью модели

$$\mathbf{s}_\psi, \mathbf{S}_\varphi = \mathbb{T}(\mathbf{x}; \mathbf{w})$$

$$\arg\max_{\mathbf{w}} \sum_q \mathbf{v}_{\mathbf{x}'}^\top \kappa(\mathbf{s}_\psi, \mathbf{S}_\varphi) \mathbf{v}_\mathbf{x}$$

Neural Logic Inductive Learning

В таком виде путь в многошаговом рассуждении можно интерпретировать как решение проблемы LP с помощью цепочки правил FOL:

$$P^*(X, X') \leftarrow P^{(1)}(X, Y_1) \wedge P^{(2)}(Y_1, Y_2) \wedge ... \wedge P^{(T)}(Y_{n-1}, X')$$

Neural Logic Inductive Learning

$$\mathbf{v}_{\mathbf{x}'}^{\top} \prod_{t=1}^{T} \mathbf{M}^{(t)} \cdot \mathbf{v}_{\mathbf{x}}$$

Chain-like

$\boldsymbol{\varphi}$

$$\psi_k(\mathbf{x}, \mathbf{x}') = \begin{cases} \sigma((\mathbf{M}_k\ \mathbf{1})^{\top}(\prod_{t'=1}^{T'} \mathbf{M}^{(t')}\mathbf{v}_{\mathbf{x}'})) & \text{if } k \in \mathcal{U}, \\ \sigma((\mathbf{M}_k \prod_{t=1}^{T} \mathbf{M}^{(t)}\mathbf{v}_{\mathbf{x}})^{\top}(\prod_{t'=1}^{T'} \mathbf{M}^{(t')}\mathbf{v}_{\mathbf{x}'})) & \text{if } k \in \mathcal{B}, \end{cases}$$

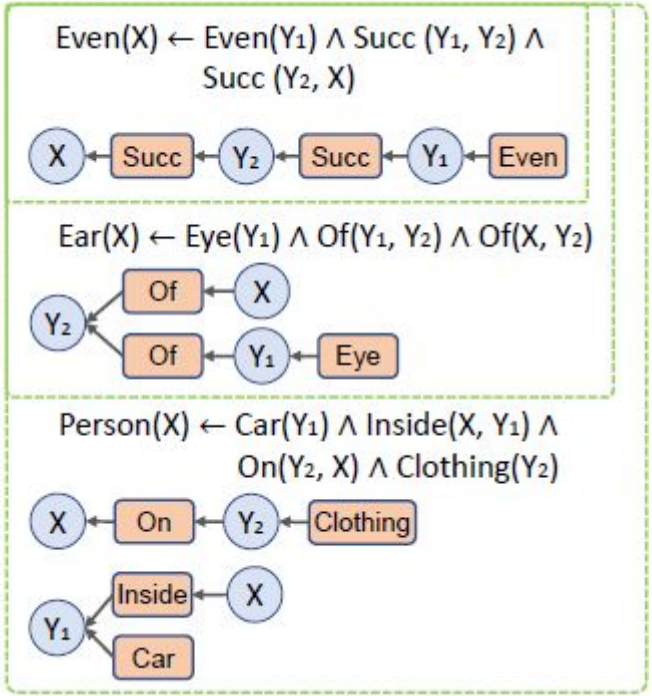
Tree-like

$\boldsymbol{\psi}$

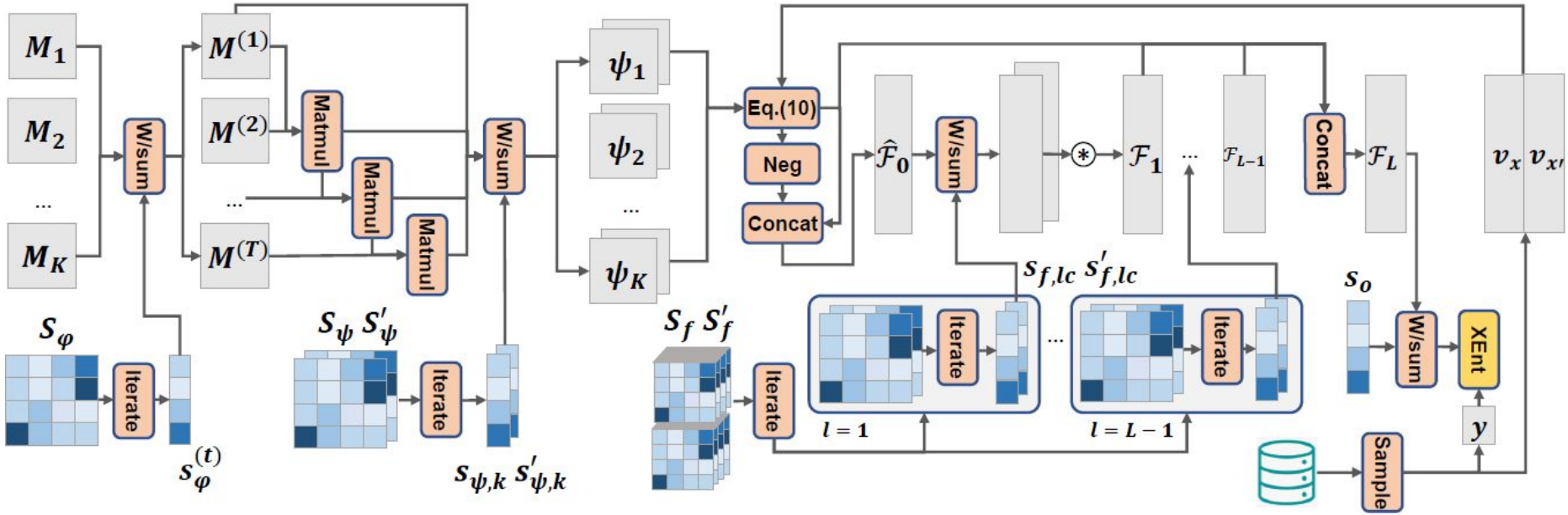
$$\mathcal{F}_0 = \Psi,\ \ \Psi = \{\psi_k(\mathbf{x}, \mathbf{x}')\}_{k=1}^{K}$$
$$\hat{\mathcal{F}}_{l-1} = \mathcal{F}_{l-1} \cup \{1 - f(\mathbf{x}, \mathbf{x}') : f \in \mathcal{F}_{l-1}\},$$
$$\mathcal{F}_l = \{f_i(\mathbf{x}, \mathbf{x}') * f_i'(\mathbf{x}, \mathbf{x}') : f_i, f_i' \in \hat{\mathcal{F}}_{l-1}\}_{i=1}^{C}$$

Conjunctions

$\boldsymbol{f}$

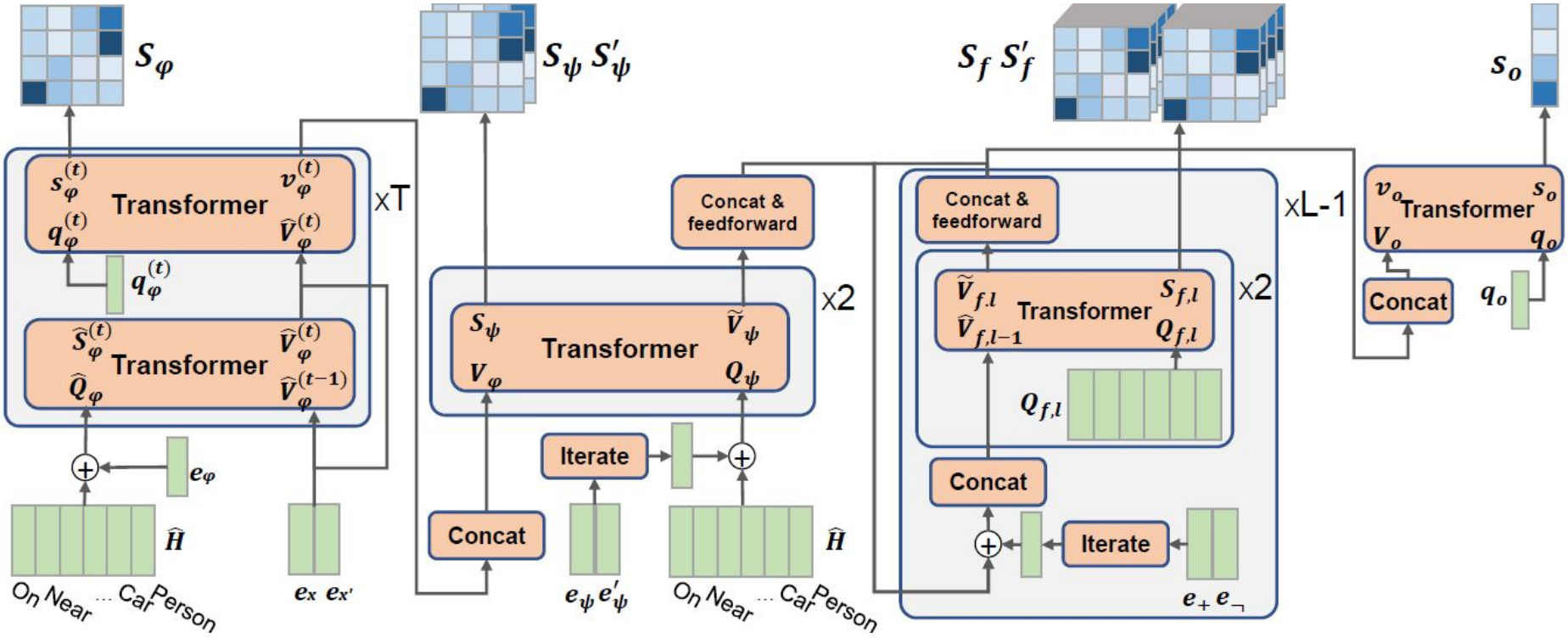


Neural Logic Inductive Learning



Пространство правил, в котором вызовы операторов, оценки операторов и логические комбинации сводятся к весовым суммам по отношению к attention векторам $\mathbf{S}_\varphi, \mathbf{S}_\psi, \mathbf{S}'_\psi, \mathbf{S}_f, \mathbf{S}'_f$ и s0. W/sum обозначает взвешенную сумму, Matmul обозначает матричное произведение, Neg обозначает not, XEnt обозначает cross-entropy loss.

Neural Logic Inductive Learning



Abductive logic program

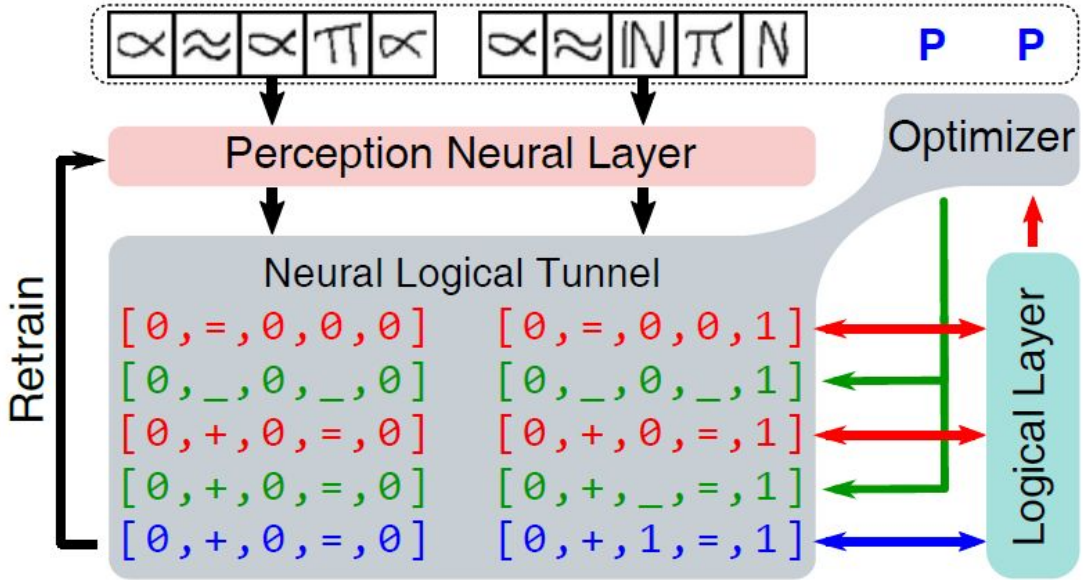
Definition 1 *Given an abductive logic theory* $T = (KB, A, IC)$*, an abductive explanation for observed data* X*, is a set,* Δ*, of ground abducibles of* A*, such that:*

- $KB \cup \Delta \models X$
- $KB \cup \Delta \models IC$
- $KB \cup \Delta$ *is consistent.*

where $\models$ *denotes the logical entailment relation.*

Абдуктивное объяснение Δ служит гипотезой, объясняющей, как наблюдение X может иметь место в соответствии с логической теорией T.

Абдуктивное обучение (Abductive Learning)



Пример использования логического абдукции для обучения слоя восприятия. Во-первых, нейронный слой восприятия неправильно интерпретирует два изображения положительных примеров и передает их в Neural Logical Tunnel (черные стрелки вниз). Затем логический уровень обнаруживает их несогласованность (красные стрелки) и отправляет запрос оптимизации на замену символов в переменных _ (зеленые стрелки). Наконец, логический уровень успешно получает согласованную интерпретацию двух изображений (синяя стрелка) и использует их в качестве меток для переобучения нейронного слоя восприятия (черная стрелка вверх).
KB: eq --> digits,[+],digits,[=],digits, пример: [1,+,1,0,1,=,1,1,0], абдуктивные утверждения - правила сложения