Wireless AI Laboratory

Relational Multi-task Learning: Modeling Relations between Data and Tasks

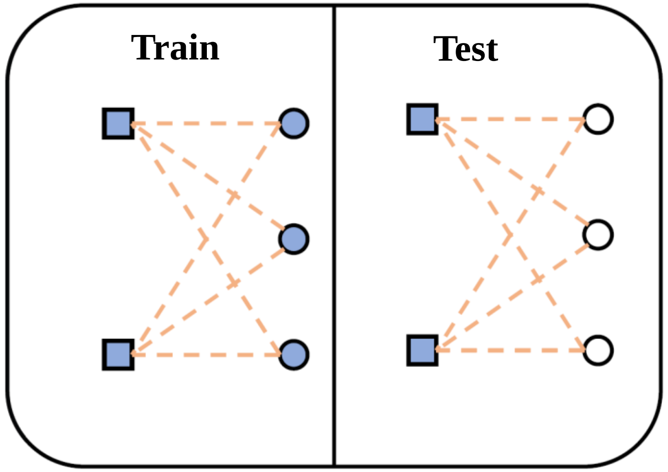
Presented by M.-D. Nguyen
Pusan National University

May 4th, 2023

1. Preliminaries: *Standard Supervised & Relational multi-task*

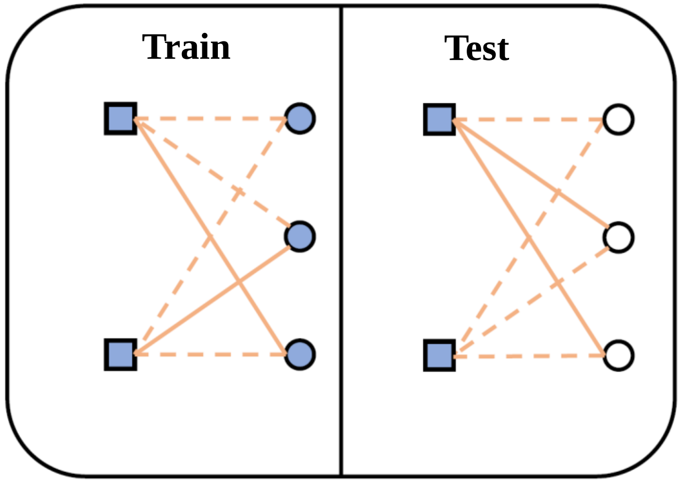
Standard Supervised:

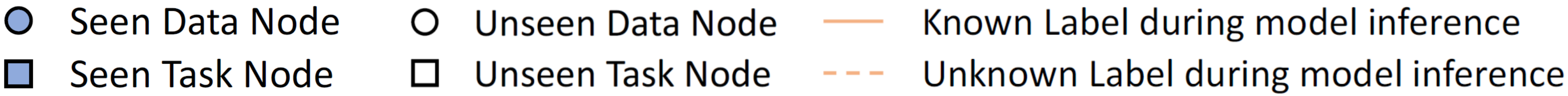
- Concept:
  - Train: $(\boldsymbol{x}^{(i)}, \{\boldsymbol{y}_j\}_{j\sim T}) \rightarrow \{\widehat{\boldsymbol{y}}_j \sim \boldsymbol{t}_j\}_{j\sim T}$
  - Test: $(\boldsymbol{x}^{(i)}, \{\boldsymbol{y}_j\}_{j\sim T}) \rightarrow \{\widehat{\boldsymbol{y}}_j \sim \boldsymbol{t}_j\}_{j\sim T}$
- Every **task** has **access** to input **data**.
- Every **task** can be operated **independently**.



Relational Multi-task:

- Concept:
  - Train: $(\boldsymbol{x}^{(i)}, \{\boldsymbol{y}_j\}_{j\sim T_{\text{aux}}}) \rightarrow \{\widehat{\boldsymbol{y}}_j \sim \boldsymbol{t}_j\}_{j\sim T_{test}}$
  - Test: $(\boldsymbol{x}^{(i)}, \{\boldsymbol{y}_j\}_{j\sim T_{\text{aux}}}) \rightarrow \{\widehat{\boldsymbol{y}}_j \sim \boldsymbol{t}_j\}_{j\sim T_{test}}$
- $\boldsymbol{T}_{aux}$ has access to data.
- $\boldsymbol{T}_{test}$ has no access to data.
- From given tasks $\boldsymbol{T}_{aux}$, we need to predict $\boldsymbol{T}_{test}$

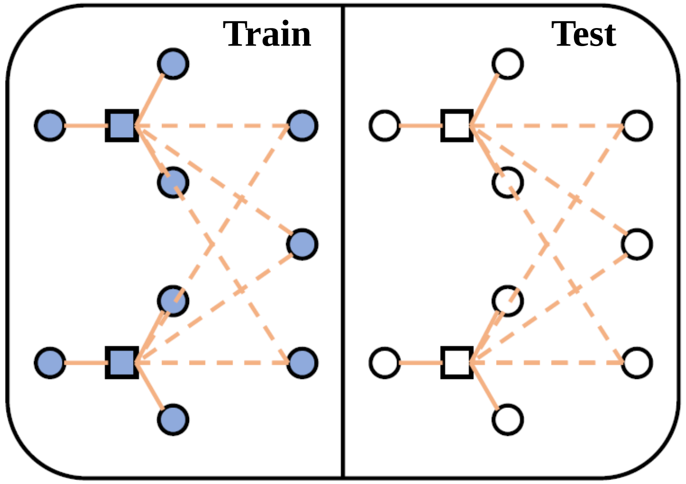




1. Preliminaries: *Meta & Relational-Meta*

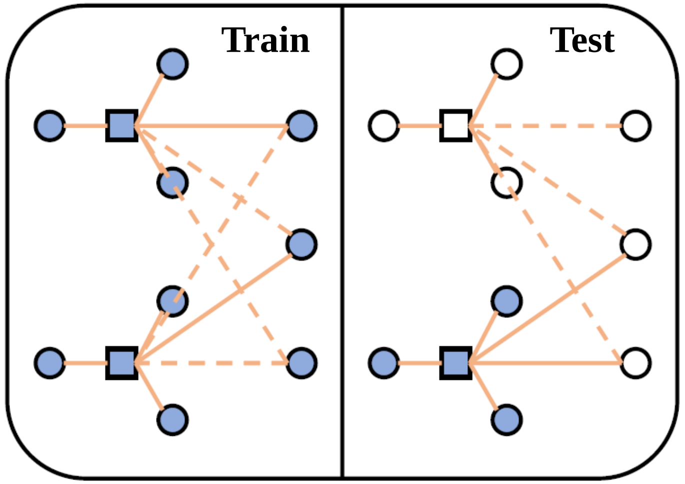
Meta Learning:

- Concept:
  - Train: $\left(x^{(i)}, \{y_j\}_{j\sim T_s}\right) \rightarrow \{\hat{y}_j \sim t_j\}_{j\sim T_s}$
  - Test: $\left(x^{(i)}, \{y_j\}_{j\sim T_u}\right) \rightarrow \{\hat{y}_j \sim t_j\}_{j\sim T_u}$
- Train on **observed** tasks.
- Aim to predict the **un-observed** tasks.



Relational Meta:

- Concept:
  - Train: $\left(x^{(i)}, \{y_j\}_{j\sim T_{\mathrm{aux}}}\right) \rightarrow \{\hat{y}_j \sim t_j\}_{j\sim T_s \setminus T_{\mathrm{aux}}}$
  - Test: $\left(x^{(i)}, \{y_j\}_{j\sim T_{\mathrm{aux}}}\right) \rightarrow \{\hat{y}_j \sim t_j\}_{j\sim T_{\mathrm{u}}}$
- Train on **observed** tasks.
- Using to predict the **un-observed** tasks.
- Observed tasks can have **access** or **no access** to data.

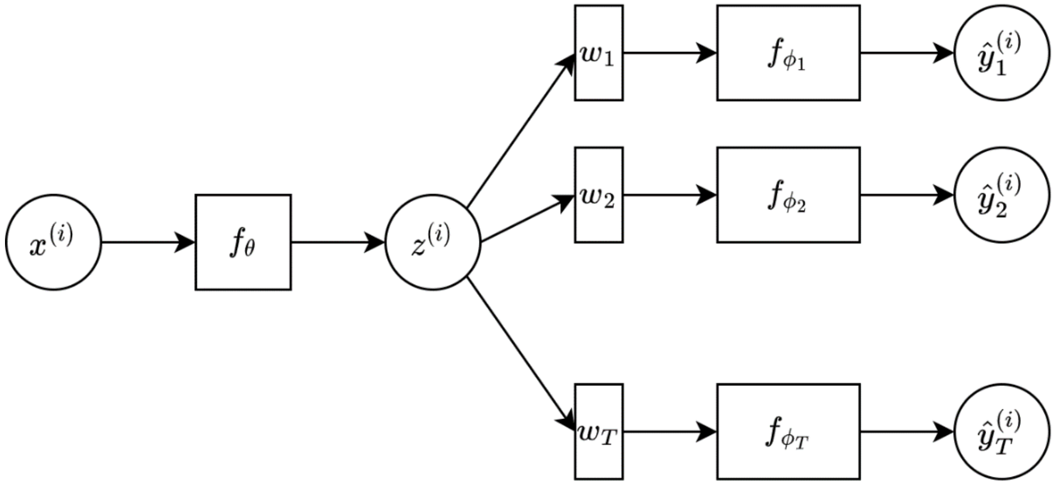


Seen Data Node
Seen Task Node
Unseen Data Node
Unseen Task Node
Known Label during model inference
Unknown Label during model inference

2. MetaLink: *Graph design*

System Model:

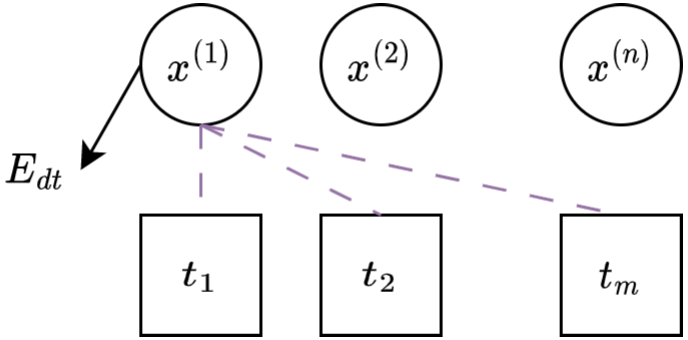
- Data:
  - $\{x^{(i)}\}$: **data** i.
  - $\{y_j^{(i)}\}_{j \in T}$: T **tasks** according to **data** i.
- Model:
  - f_θ: encoder (embedding function).
  - f_w: single weight matrix.
  - f_ϕ: task heads.



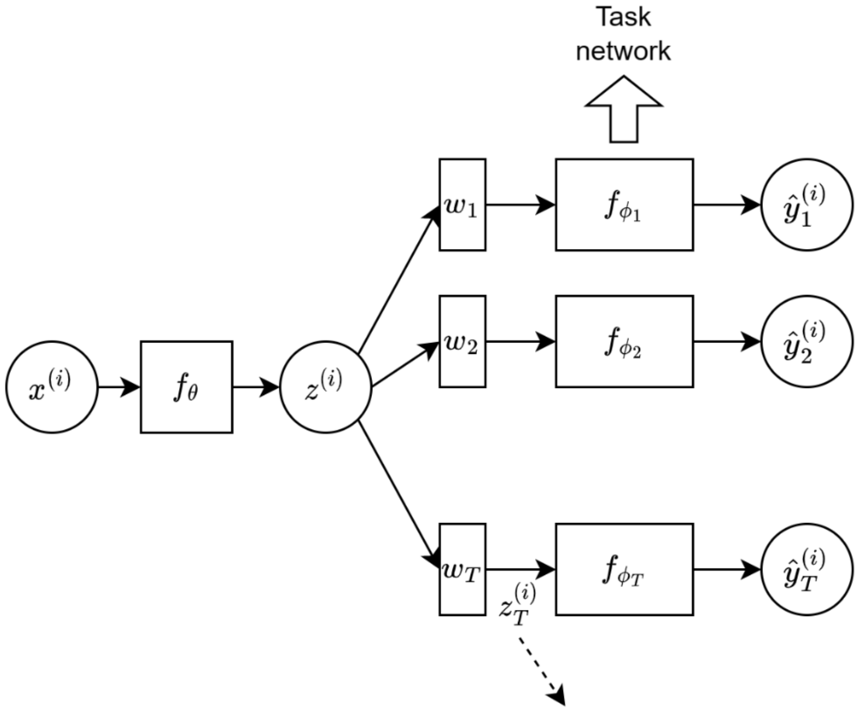
Knowledge Graph (Bi-partiate Graph):

- ***Graph***: $G = (V, E)$.
- ***Node***:
  - *Data node*: $V_d = \{x^{(1)}, x^{(2)}, \cdots, x^{(n)}\}$.
  - *Task node*: $V_t = \{t_1, t_2, \cdots, t_m\}$.
- ***Edge***:
  - *Data-task*: each pair data $x^{(i)}$ - task t_j has a link:

$$E_{dt} = \left\{\{x^{(i)}, t_j\} \sim y_{j \in T}^{(i)}\right\}$$



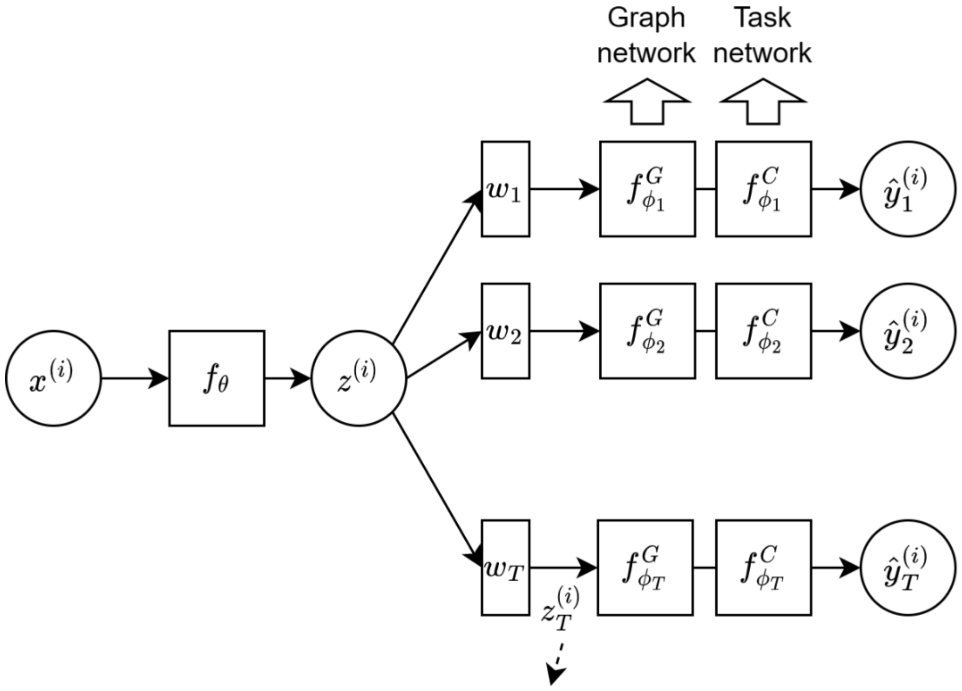
2. MetaLink: *Architecture*



Standard MTL

$$z_T^{(i)} = w_T \cdot z^{(i)}$$

Agnostic about shared **representations** and task-wise **parameters**



MetaLink

$$z_T^{(i)} = \{w_T, z^i, E(w_T, z^i)\}$$

Understand both shared **representations** and task-wise **parameters,** and their **relationship**.

2. MetaLink: *Node Initialization*

Graph:

- Node:
  - Data: $h_i^{(0)} = z^{(i)}$
  - Task:
    - $h_j^{(0)} = w^{(j)}$ if not meta task.
    - $h_j^{(0)} = 1$ if meta task.
- Edge:
  - Data-task: $E_{dt} = \left\{h_i^{(0)}, h_j^{(0)}\right\}$

3. GNN-MetaLink: *Graph embedding network*

Message Passing:

Trainable parameters

Trainable parameters

Reduce info loss

$$\boldsymbol{h}_v^{(l)} = \boldsymbol{U}^{(l)} \cdot CONCAT\left(MEAN\left(\left\{RELU\left(\boldsymbol{W}^{(l)}_{\mathbf{1}[v\in \boldsymbol{V}_d, u\in \boldsymbol{V}_t]}\boldsymbol{h}_u^{(l-1)}\right), u \in \mathcal{N}(v)\right\}\right), \boldsymbol{h}_v^{(l-1)}\right)$$

Message Passing (aggregate task output):

$$\boldsymbol{h}_v^{(l)} = \boldsymbol{U}^{(l)} \cdot CONCAT\left(MEAN\left(\left\{RELU\left(\boldsymbol{W}^{(l)}_{\mathbf{1}[v\in \boldsymbol{V}_d, u\in \boldsymbol{V}_t]}\boldsymbol{h}_u^{(l-1)} + \boldsymbol{O}^{(l)} y_v^{(u)}\right), u \in \mathcal{N}(v)\right\}\right), \boldsymbol{h}_v^{(l-1)}\right)$$

3. GNN-MetaLink: *Edge Predictor*

Output:

Aggregate task node, data node

Classifier $\hat{y}_j^i = MLP\left(CONCAT\left(h_j^{(L)}, h_i^{(L)}\right)\right)$

Classifier

3. GNN-MetaLink: *Relational Training Process*

Algorithm 2 MetaLink Training in Relational Setting

Require: Dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, y)\}$. A parameterized embedding function f_θ. Last layer weights for each task $\{\mathbf{w}_j\}$. A parameterized heterogeneous GNN f_ϕ. Number of GNN layers L.

for each iteration **do**

$\{(\mathbf{x}, \{y^{(i)}_{j\in T^{(i)}_{\text{aux}}}\}, \{y^{(i)}_{j\in T^{(i)}_{\text{test}}}\}\} \leftarrow \text{SampleMiniBatch}(\mathcal{D}_{\text{train}})$ → Graph size depends on batch size

$\{\mathbf{z}\} \leftarrow f_\theta(\mathbf{x})$ for $\mathbf{x} \in \{(\mathbf{x}, \{y^{(i)}_{j\in T^{(i)}_{\text{aux}}}\}, \{y^{(i)}_{j\in T^{(i)}_{\text{test}}}\}\}$

$V_d^{(0)} = \{\mathbf{h}_i^{(0)} \leftarrow \mathbf{z}$ for $\mathbf{z} \in \{\mathbf{z}\}\}$ ▷ Initialize data nodes

$V_t^{(0)} = \{\mathbf{h}_j^{(0)} \leftarrow \mathbf{w}_j$ for $\mathbf{w}_j \in \{\mathbf{w}_j\}\}$ ▷ Initialize task nodes

$E = \{\mathbf{e}_{ij} \leftarrow (\mathbf{x}^{(i)}, t_j)$ for $y_j^{(i)} \in \{y^{(i)}_{j\in T_{\text{aux}}}\}\}$ ▷ Initialize edges

for $l = 1$ to L **do**

$V_d^{(l)}, V_t^{(l)} \leftarrow \text{GraphConv}(V_d^{(l-1)}, V_t^{(l-1)}, E)$ with f_ϕ

logits $\leftarrow \text{EdgePred}(V_d^{(L)}, V_t^{(L)})$ with f_ϕ

Backward $\Big(\text{Criterion}(\text{logits}, \{\{y_j^{(i)}\}_{j\in T^{(i)}_{\text{test}}}\})\Big)$

3. GNN-MetaLink: *Training Process*

Algorithm 3 MetaLink Training in Meta Setting

Require: Dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, y)\}$. A parameterized embedding function f_θ. A parameterized heterogeneous GNN f_ϕ. Number of GNN layers L.

for each iteration **do**
  $S, Q \leftarrow \text{SampleMiniBatch}(\mathcal{D}_{\text{train}})$ ▷ Simulate meta setting in training
  $\{\mathbf{z}\} \leftarrow f_\theta(\mathbf{x})$ for $\mathbf{x} \in (S, Q)$
  $V_d^{(0)} = \{\mathbf{h}_i^{(0)} \leftarrow \mathbf{z}$ for $\mathbf{z} \in \{\mathbf{z}\}\}$ ▷ Initialize data nodes
  $V_t^{(0)} = \{\mathbf{h}_j^{(0)} \leftarrow \mathbf{1}\}$ ➡ All tasks are meta ▷ Initialize task nodes
  $E = \{\mathbf{e}_{ij} \leftarrow (\mathbf{x}^{(i)}, t_j)$ for $y_j^{(i)} \in S\}$ ➡ edges of support set ▷ Initialize edges
  for $l = 1$ to L **do**
    $V_d^{(l)}, V_t^{(l)} \leftarrow \text{GraphConv}(V_d^{(l-1)}, V_t^{(l-1)}, E)$ with f_ϕ
  logits $\leftarrow \text{EdgePred}(V_d^{(L)}, V_t^{(L)})$ with f_ϕ
  Backward $\left(\text{Criterion}(\text{logits}, \{\{y_j^{(i)}\}_{j \in T_s}\} \in Q)\right)$ ➡ SGD on query set

3. GNN-MetaLink: *Training Process*

Algorithm 1 MetaLink Training in Relational Meta Setting

Require: Dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, y)\}$. A parameterized embedding function f_θ. Last layer weights for each task $\mathbf{w}_j$. A parameterized heterogeneous GNN f_ϕ. Number of GNN layers L.

for each iteration **do**
  $S, Q \leftarrow \text{SampleMiniBatch}(\mathcal{D}_{\text{train}})$ ▷ Simulate meta setting in training
  $\{\mathbf{z}\} \leftarrow f_\theta(\mathbf{x})$ for $\mathbf{x} \in (S, Q)$
  $V_d^{(0)} = \{\mathbf{h}_i^{(0)} \leftarrow \mathbf{z}$ for $\mathbf{z} \in \{\mathbf{z}\}\}$ ▷ Initialize data nodes
  $V_t^{(0)} = \{\mathbf{h}_j^{(0)} \leftarrow \mathbf{1}$ if meta else $\mathbf{w}_j$ for each $\mathbf{w}_j\}$ ▷ Initialize task nodes
  $E = \{\mathbf{e}_{ij} \leftarrow (\mathbf{x}^{(i)}, t_j)$ for $y_j^{(i)} \in (S, Q)\}$ ▷ Initialize edges
  for $l = 1$ to L **do**
    $V_d^{(l)}, V_t^{(l)} \leftarrow \text{GraphConv}(V_d^{(l-1)}, V_t^{(l-1)}, E)$ with f_ϕ
  logits $\leftarrow \text{EdgePred}(V_d^{(L)}, V_t^{(L)})$ with f_ϕ
  $\text{Backward}\left(\text{Criterion}(\text{logits}, \{\{y_j^{(i)}\}_{j \in T_s^{(i)} \setminus T_{\text{aux}}^{(i)}}\} \in Q)\right)$

generalize to meta tasks

Train only on unseen tasks

Thank you