Automatic Domain Adaptation by Transformers in In-Context Learning

https://arxiv.org/abs/2405.16819

Ryuichiro Hataya[α], **Kota Matsui**[β, α], **Masaaki Imaizumi**[γ, α]

[α] RIKEN AIP, [β] Nagoya University, [γ] The University of Tokyo

contact: ryuichiro.hataya@riken.jp / webpage: hataya.tokyo

Summary

- Selecting or designing suitable UDA algorithms for given problems is challenging

- We showed that Transformers can
  - approximate UDA algorithms (IWL and DANN)
  - select appropriate one based on data statistics

  in the ICL framework

Unsupervised Domain Adaptation (UDA)

Source
labeled data

$$\mathcal{D}_S = \{(\boldsymbol{x}_i^S, y_i^S)\}_{i=1}^{n}$$
$$(\boldsymbol{x}_i^S, y_i^S) \sim p_S$$

Target
unlabeled data

$$\mathcal{D}_T = \{\boldsymbol{x}_i^T\}_{i=1}^{n'}$$
$$(\boldsymbol{x}_i^T, \cdot) \sim p_T$$

transfer “knowledge” of source
to classify target data

Goal: minimize the target risk

i.e., get $\underset{f \in \mathcal{F}}{\operatorname{argmin}} R(f)$, where $R(f) = \mathbb{E}_{(\boldsymbol{x}, y) \sim p_T}[\ell(f(\boldsymbol{x}), y)]$

Unavailable!

Images: MNIST (left) and SVHN (right)

UDA: Instance-based Approach

When $\mathcal{X}_S \times \mathcal{Y}_S = \mathcal{X}_T \times \mathcal{Y}_T$ and $p_S(\boldsymbol{x}, y) \neq p_T(\boldsymbol{x}, y)$

Assume **covariate shift** $p_S(y|\boldsymbol{x}) = p_T(y|\boldsymbol{x})$

Importance-weighted learning [Sugiyama+12, Kimura&Hino24]

$$R(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_T}[\ell(f(\boldsymbol{x}, y))] = \mathbb{E}_{(\boldsymbol{x},y)\sim p_S}[q(\boldsymbol{x})\ell(f(\boldsymbol{x}), y)]$$, where $q(\boldsymbol{x}) = \frac{p_T(\boldsymbol{x})}{p_S(\boldsymbol{x})}$

1. Estimate density ratio by e.g., uLSIF [Kanamori+09]

$$\hat{q}_{\hat{\boldsymbol{\alpha}}} = \hat{\boldsymbol{\alpha}}^\top \boldsymbol{\phi}(\boldsymbol{x})$$, where $\hat{\boldsymbol{\alpha}} = \operatorname*{argmin}_{\boldsymbol{\alpha}} \frac{1}{2} \int (\alpha^\top \boldsymbol{\phi}(\boldsymbol{x}) - q(\boldsymbol{x}))^2 p_S(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}$

2. Minimize the target risk

$$\hat{R}(\boldsymbol{w}) = \sum_{(\boldsymbol{x},y)\in\mathcal{D}_S} \hat{q}_{\hat{\boldsymbol{\alpha}}}(\boldsymbol{x}) \ell(\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}), y)$$

UDA: Feature-based Approach

When $\mathcal{X}_S \times \mathcal{Y}_S \neq \mathcal{X}_T \times \mathcal{Y}_T$

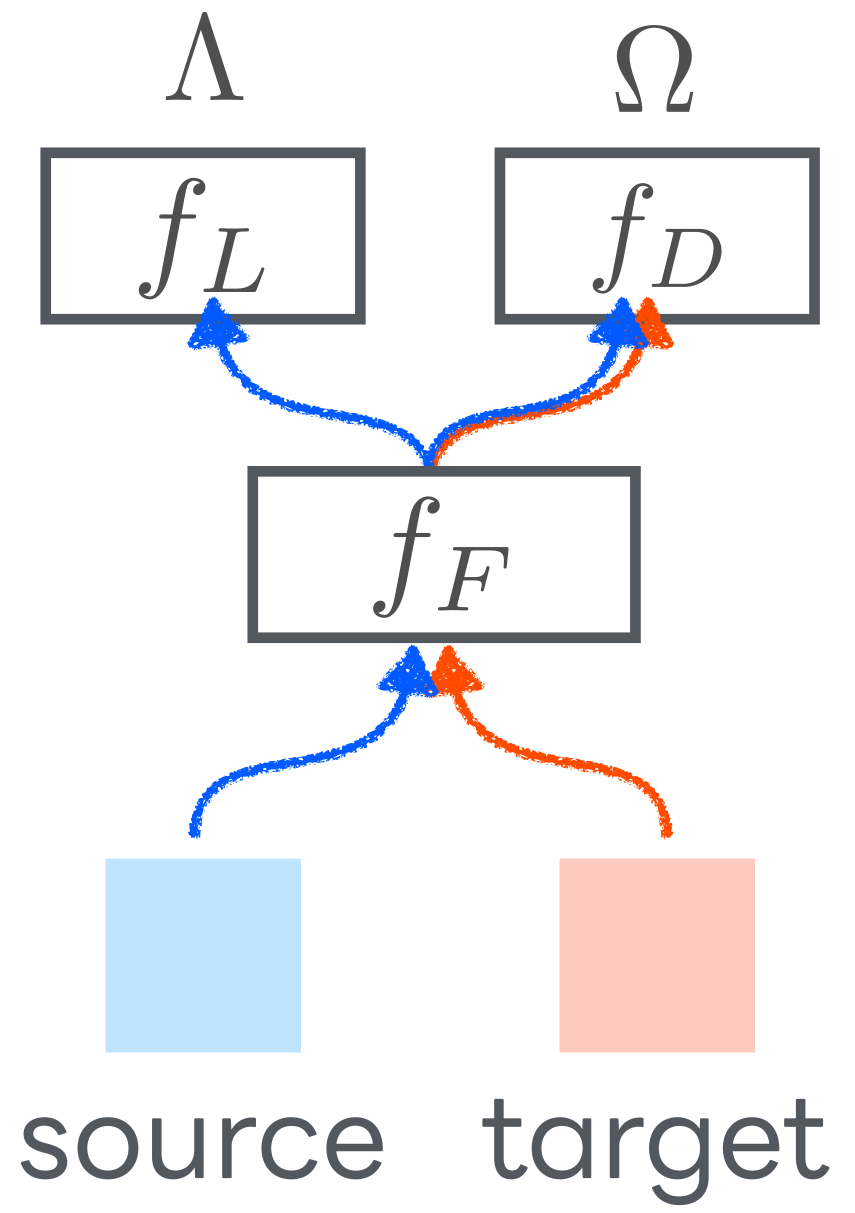
Find domain invariant features ψ s.t. $R(f') \approx \mathbb{E}_{(\boldsymbol{x},y)\sim p_S}[\ell(f'(\boldsymbol{\psi}(\boldsymbol{x})), y)]$

Domain Adversarial Neural Networks (DANNs) [Ganin+16]

f_F : feature extractor, parameterized by $\boldsymbol{\theta}_F$ (ψ)

f_L : label classifier, parameterized by $\boldsymbol{\theta}_L$ (f')

f_D : domain classifier, parameterized by $\boldsymbol{\theta}_D$



DANN obtains ψ and f' by adversarial learning

$$\min_{\boldsymbol{\theta}_F, \boldsymbol{\theta}_L} \max_{\boldsymbol{\theta}_D} \Lambda(\boldsymbol{\theta}_F, \boldsymbol{\theta}_L) - \lambda\Omega(\boldsymbol{\theta}_F, \boldsymbol{\theta}_D)$$

In-context Learning (ICL)

LLMs perform new tasks from instructions in prompts

cat: 0
dog: 0
whale: 1
dolphin: 1
cod: 0
salmon: 1

training data

eel: ?

test data

ChatGPT

0

→ LLMs perform “machine learning” without parameter update

ICL: Theory

Appropriately pre-trained TFs can perform ML algorithms

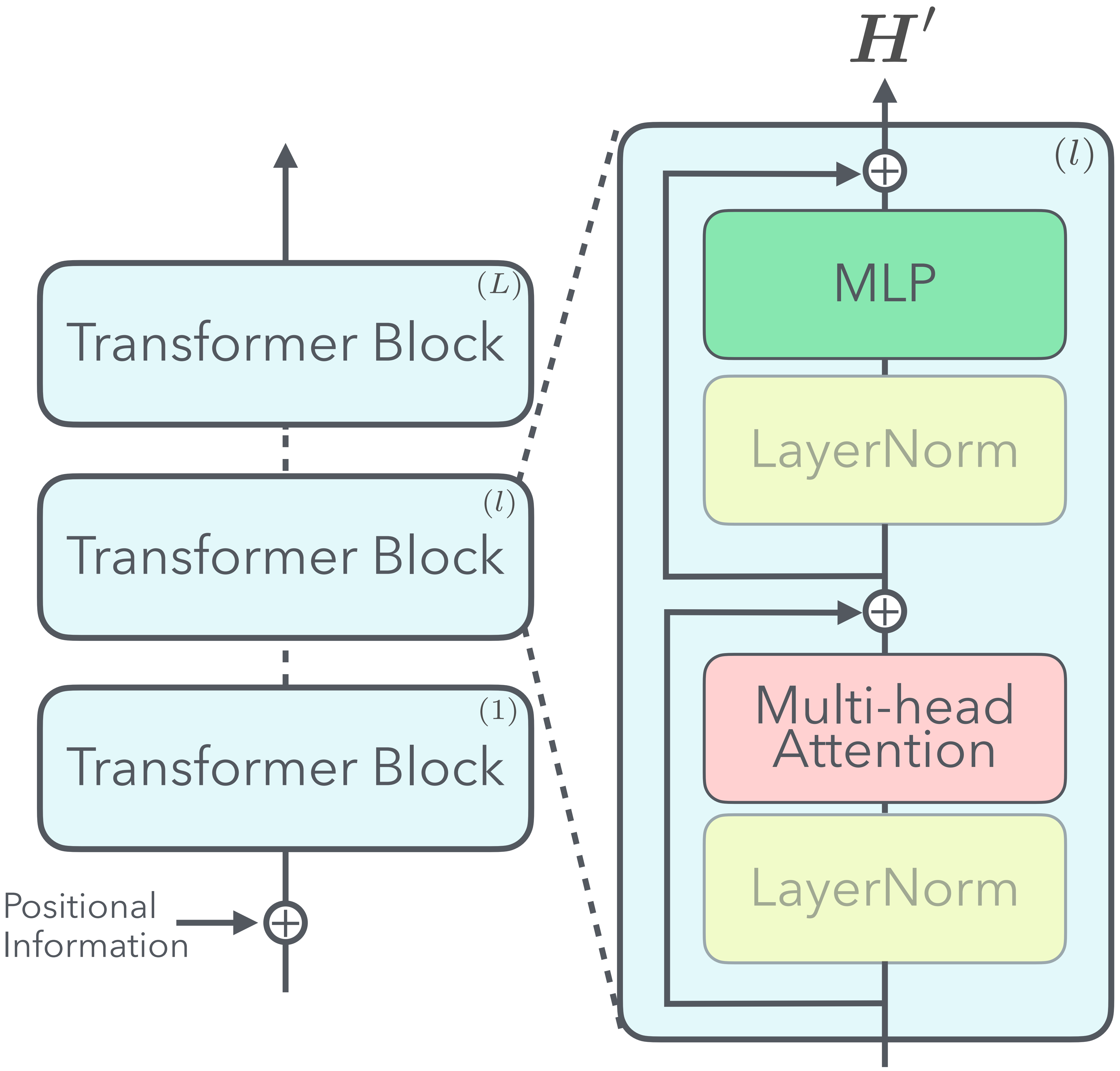
- Least squares [Zhang+23, Akyürek+23]
- Gradient descent of linear regression [von Oswald+22, Akyürek+23]

  GLMs / neural networks [Bai+23]
- Linear UCB, Thompson sampling, ⋯ [Lin+24]

TFs can also perform model selection [Bai+23]

Based on validation performance / data types

Transformer



$$\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n, \ldots, \boldsymbol{h}_N] \in \mathbb{R}^{N \times D}$$

token

$$\mathrm{MLP}^{(l)}(\boldsymbol{H}) = \boldsymbol{H} + \boxed{\boldsymbol{W}_2^{(l)} \varsigma(\boldsymbol{W}_1^{(l)} \boldsymbol{H})}$$

intra-token transformation

$\boldsymbol{W}_1^{(l)} \in \mathbb{R}^{D' \times D}, \boldsymbol{W}_2^{(l)} \in \mathbb{R}^{D \times D'}$: learnable parameters

$$\mathrm{Attn}^{(l)}(\boldsymbol{H}) = \boldsymbol{H} + \boxed{\frac{1}{N} \sum_{m=1}^{M} \boldsymbol{V}_m^{(l)} \boldsymbol{H} \sigma((\boldsymbol{Q}_m^{(l)} \boldsymbol{H})^\top \boldsymbol{K}_m^{(l)} \boldsymbol{H})}$$

inter-token transformation

$\boldsymbol{K}_m^{(l)}, \boldsymbol{Q}_m^{(l)}, \boldsymbol{V}_m^{(l)} \in \mathbb{R}^{D \times D}$: learnable parameters

ς σ : activation function (ReLU in this work)

ICL Framework by Bai+23

1. Sum-of-ReLUs can approximate any functions

e.g., $f(x,y) = \sum_{m=1}^{M} c_m \sigma(a_m x + b_m y + d_m)$ can approximate any bivariate functions

2. Multi-head Attention can represent any aggregation functions

$$\mathrm{Attn}^{(l)}(\boldsymbol{h}_i) = \boldsymbol{h}_i + \underbrace{\frac{1}{N}\sum_{j=1}^{N} \underbrace{\sum_{m=1}^{M} \sigma(\langle \boldsymbol{Q}_m^{(l)} \boldsymbol{h}_i, \boldsymbol{K}_m^{(l)} \boldsymbol{h}_j \rangle) \boldsymbol{V}_m^{(l)} \boldsymbol{h}_j}_{\text{approximate a function}}}_{\text{aggregation}}$$

aggregation
e.g., mean of selected data

approximate a function

js with positive values in σ are selected

3. Single attention block can imitate a single GD step of GLM

Let $\boldsymbol{h}_i = \begin{bmatrix} \boldsymbol{x}_i \\ y_i \\ \boldsymbol{w} \end{bmatrix}$

$$\hat{L} = \sum_{j=1}^{N} \ell(\boldsymbol{w}^\top \boldsymbol{x}_j, y_j)$$

model parameter
loss function

$$\mathrm{Attn}^{(l)}(\boldsymbol{h}_i) = \begin{bmatrix} \boldsymbol{x}_i \\ y_i \\ \boldsymbol{w} \end{bmatrix} + \underbrace{\frac{1}{N}\sum_{j=1}^{N} \underbrace{\sum_{m=1}^{M} \sigma(\overbrace{\langle \boldsymbol{Q}_m^{(l)} \boldsymbol{h}_i, \boldsymbol{K}_m^{(l)} \boldsymbol{h}_j \rangle}^{a_m \boldsymbol{w}^\top \boldsymbol{x}_j + b_m y_j + d_m}) \overbrace{\boldsymbol{V}_m^{(l)} \boldsymbol{h}_j}^{-\eta c_m \begin{bmatrix} \boldsymbol{0} \\ 0 \\ \boldsymbol{x}_j \end{bmatrix}}}_{\approx -\eta \partial_1 \ell(\boldsymbol{w}^\top \boldsymbol{x}_j, y_j) \begin{bmatrix} \boldsymbol{0} \\ 0 \\ \boldsymbol{x}_j \end{bmatrix}}}_{\approx \begin{bmatrix} \boldsymbol{0} \\ 0 \\ -\eta \nabla_w \hat{L} \end{bmatrix}}$$

ICL Framework by Bai+23

3. Single attention block can imitate a single GD step of GLM

Let $\boldsymbol{h}_i = \begin{bmatrix} \boldsymbol{x}_i \\ y_i \\ \boldsymbol{w} \end{bmatrix}$

$$\hat{L} = \sum_{j=1}^{N} \ell(\boldsymbol{w}^\top \boldsymbol{x}_j, y_j)$$

loss function: ℓ; model parameter: $\boldsymbol{w}$

$$\mathrm{Attn}^{(l)}(\boldsymbol{h}_i) = \begin{bmatrix} \boldsymbol{x}_i \\ y_i \\ \boldsymbol{w} \end{bmatrix} + \underbrace{\frac{1}{N}\sum_{j=1}^{N} \underbrace{\sum_{m=1}^{M} \sigma(\underbrace{\langle \boldsymbol{Q}_m^{(l)} \boldsymbol{h}_i, \boldsymbol{K}_m^{(l)} \boldsymbol{h}_j \rangle}_{a_m \boldsymbol{w}^\top \boldsymbol{x}_j + b_m y_j + d_m}) \underbrace{\boldsymbol{V}_m^{(l)} \boldsymbol{h}_j}_{-\eta c_m \begin{bmatrix} \mathbf{0} \\ 0 \\ \boldsymbol{x}_j \end{bmatrix}}}_{\approx -\eta \partial_1 \ell(\boldsymbol{w}^\top \boldsymbol{x}_j, y_j) \begin{bmatrix} \mathbf{0} \\ 0 \\ \boldsymbol{x}_j \end{bmatrix}}}_{\approx \begin{bmatrix} \mathbf{0} \\ 0 \\ -\eta \nabla_w \hat{L} \end{bmatrix}}$$

4. L-layer TF can approximate L steps of GD

Let the input be $\boldsymbol{H}^{(1)} = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \dots & \boldsymbol{x}_n & \boldsymbol{x}_* \\ y_1 & y_2 & \dots & y_n & 0 \\ 1 & 1 & \dots & 1 & 1 \\ 1 & 1 & \dots & 1 & 0 \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \end{bmatrix}$

Marking training data

training data test data

For any $\varepsilon > 0$, there exists an L-layer TF s.t. $\|\mathtt{read}(\mathrm{TF}(\boldsymbol{H}^{(1)})) - (\boldsymbol{w}^*)^\top \boldsymbol{x}_*\| \leq \varepsilon$

model parameter after L steps of GD

Setup: In-context Learning Domain Adaptation

We encode data as

$$
\boldsymbol{H}^{(1)} = \begin{bmatrix} \boldsymbol{x}_1^S & \dots & \boldsymbol{x}_n^S & \boldsymbol{x}_1^T & \dots & \boldsymbol{x}_{n'}^T & \boldsymbol{x}_* \\ y_1 & \dots & y_n & 0 & \dots & 0 & 0 \\ 1 & \dots & 1 & 0 & \dots & 0 & 0 \\ 1 & \dots & 1 & 1 & \dots & 1 & 1 \\ 1 & \dots & 1 & 1 & \dots & 1 & 1 \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \end{bmatrix}
$$

Marking data from source: d_i

Marking training data

training data from source

training data from target

test data from target

Note: inputs like $[\boldsymbol{x}_1^S, y_1, \boldsymbol{x}_2^S, y_2, \dots, \boldsymbol{x}_1^T, \dots, \boldsymbol{x}_*]$

can be converted into $\boldsymbol{H}^{(1)}$ using some TF layers

Main results: ICL Domain Adaptation Algorithms

Theorem 1

For any $\varepsilon > 0$, there exists a $2L+1$ -layer Transformer that can approximate importance-weighted learning with the uLSIF estimator:

$$\|\mathtt{read}(\mathrm{TF}(\boldsymbol{H}^{(1)})) - \hat{f}^{\mathrm{IWL}}(\boldsymbol{x}_*)\| \leq \varepsilon$$

Theorem 2

For any $\varepsilon > 0$, there exists a $2L$ -layer Transformer that can approximate a two-layer domain adversarial neural network:

$$\|\mathtt{read}(\mathrm{TF}(\boldsymbol{H}^{(1)})) - \hat{f}^{\mathrm{DANN}}(\boldsymbol{x}_*)\| \leq L\varepsilon$$

Transformers can also approximate other UDA algorithms

Main results: ICL Algorithm Selection

Select the “appropriate” result for given data

🤔 No labels for target data. How to select one?

Use IWL if supports of p_S and p_T overlap sufficiently, otherwise DANN

$$\hat{f}^{\mathrm{ICUDA}}(\boldsymbol{x}_*) = \begin{cases} \hat{f}^{\mathrm{IWL}}(\boldsymbol{x}_*) & \text{if} \quad \min_{\boldsymbol{x} \sim \mathcal{D}_T} p_S(\boldsymbol{x}) > \delta \\ \hat{f}^{\mathrm{DANN}}(\boldsymbol{x}_*) & \text{otherwise} \end{cases}$$

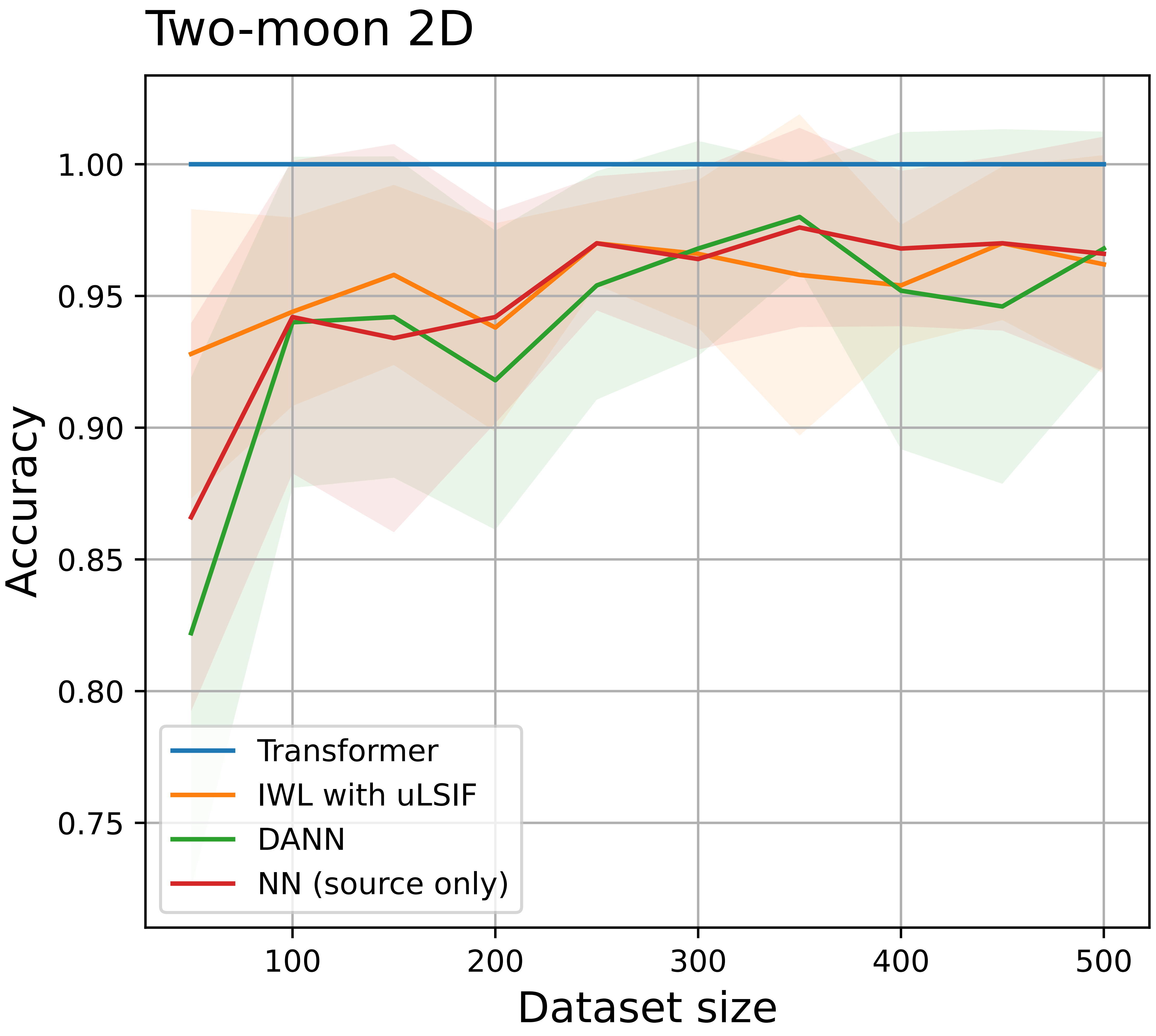
Theorem 3

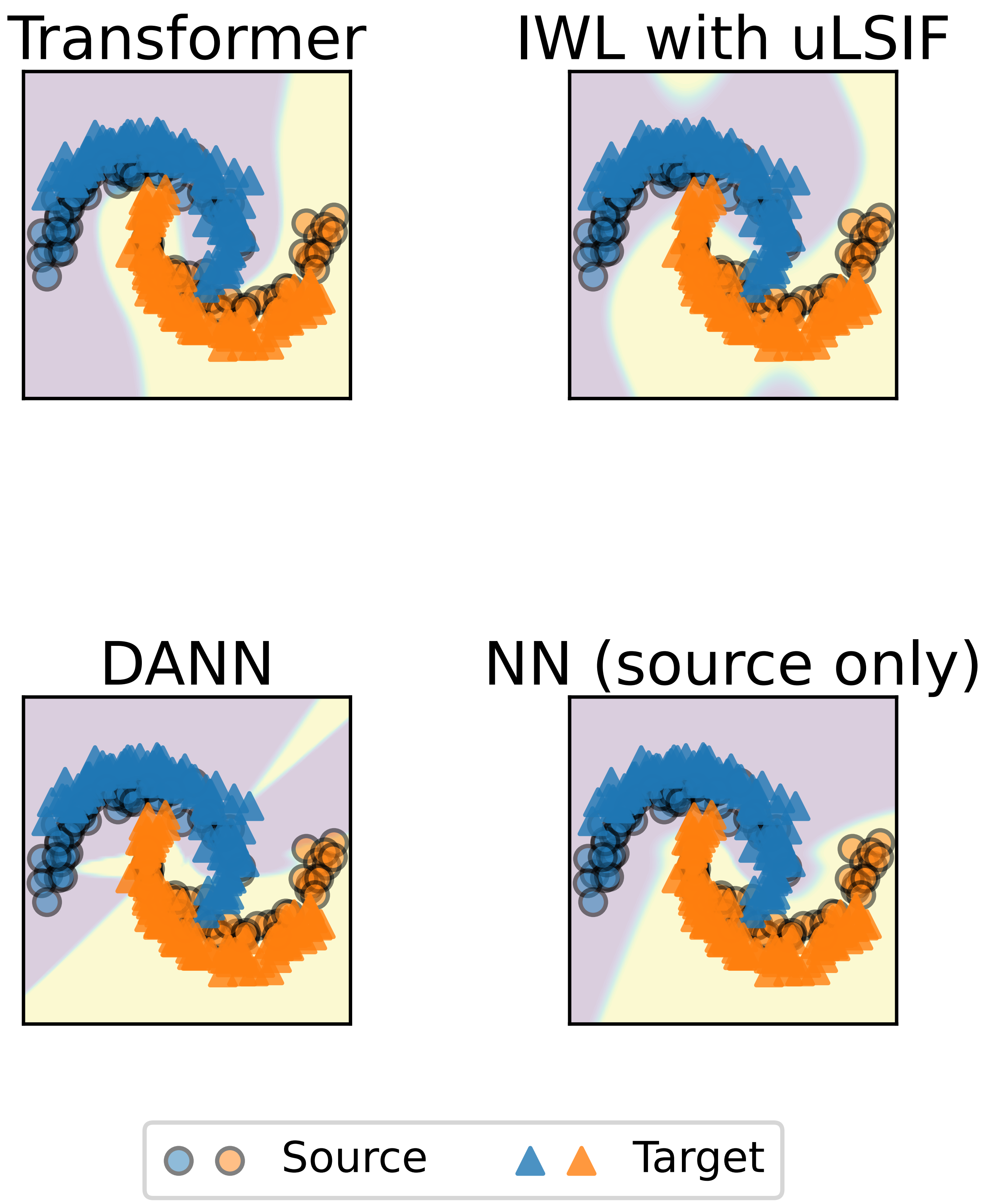
For any $\varepsilon > 0$, there exists a three-layer Transformer that approximately select the appropriate result with prob. at least $1 - \frac{1}{n'} - O(\varepsilon)$:

$$\|\mathtt{read}(\mathrm{TF}(\boldsymbol{H}')) - \hat{f}^{\mathrm{ICUDA}}(\boldsymbol{x}_*)\| \leq \varepsilon$$

Transformers can also approximate other selection rules

Experiments





Summary

- We showed that Transformers can
  - approximate UDA algorithms (IWL and DANN)
  - select appropriate one based on data statistics

  in the ICL framework

- TFs can approximate other algorithms / selection rules
  - Mixing / combining algorithms is also possible

  → LLMs may know better UDA algorithms unknown to us

contact: ryuichiro.hataya@riken.jp / webpage: hataya.tokyo