LAMARR
INSTITUTE FOR MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Domain Adaptation for Flavour Tagging at LHCb

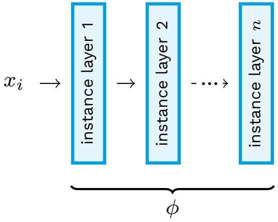
Mirko Bunse and Quentin Führung

Lamarr Physics Monthly – April 7th, 2025

Partner institutions:
technische universität dortmund
Fraunhofer IAIS
UNIVERSITÄT BONN
Fraunhofer IML

Institutionally funded by:
Bundesministerium für Bildung und Forschung
Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen

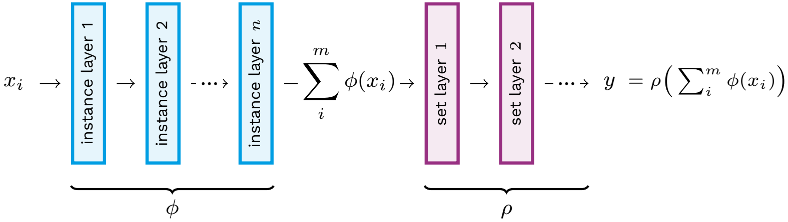
Recap: Deep Sets



- Each event is a **set of tracks** $\{x_i \in \mathcal{X} : 1 \leq i \leq m\}$ of variable size m

[1] Zaheer et al., "Deep sets", 2017
[2] Lee et al., "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks", 2019

Recap: Deep Sets

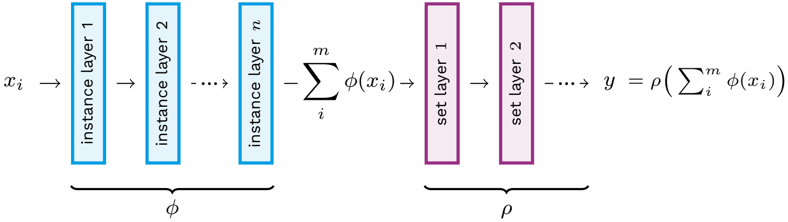


- Each event is a **set of tracks** $\{x_i \in \mathcal{X} : 1 \leq i \leq m\}$ of variable size m
- Track representations $\phi(x_i)$ are aggregated by (un-weighted) **sum pooling**

[1] Zaheer et al., “Deep sets”, 2017

[2] Lee et al., “Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks”, 2019

Recap: Deep Sets



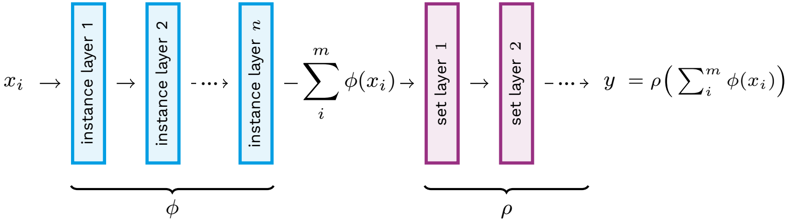
- Each event is a **set of tracks** $\{x_i \in \mathcal{X} : 1 \leq i \leq m\}$ of variable size m
- Track representations $\phi(x_i)$ are aggregated by (un-weighted) **sum pooling**

Deep Sets are **universal approximators** of permutation-invariant functions[1] (✓), but they ...

[1] Zaheer et al., "Deep sets", 2017

[2] Lee et al., "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks", 2019

Recap: Deep Sets



- Each event is a **set of tracks** $\{x_i \in \mathcal{X} : 1 \leq i \leq m\}$ of variable size m
- Track representations $\phi(x_i)$ are aggregated by (un-weighted) **sum pooling**

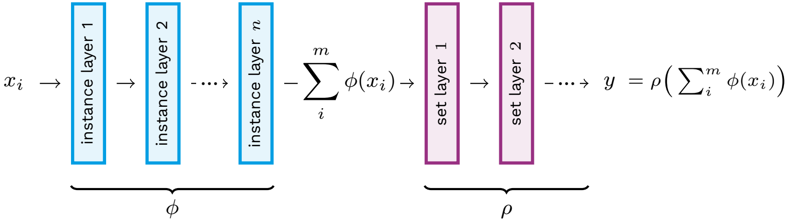
Deep Sets are **universal approximators** of permutation-invariant functions[1] (✓), but they ...

- might be inefficient in representing interactions between tracks[2] ✗

[1] Zaheer et al., "Deep sets", 2017

[2] Lee et al., "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks", 2019

Recap: Deep Sets



- Each event is a **set of tracks** $\{x_i \in \mathcal{X} : 1 \leq i \leq m\}$ of variable size m
- Track representations $\phi(x_i)$ are aggregated by (un-weighted) **sum pooling**

Deep Sets are **universal approximators** of permutation-invariant functions[1] (✓), but they ...

- might be inefficient in representing interactions between tracks[2] ✗
- might over-fit to simulated training data ✗

[1] Zaheer et al., “Deep sets”, 2017

[2] Lee et al., “Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks”, 2019

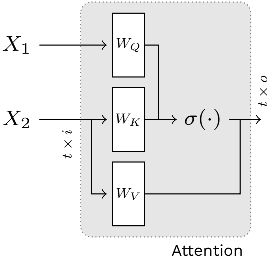
Set Transformers

Idea: Model track interactions through Self-Attention[2]

Set Transformers

Idea: Model track interactions through Self-Attention[2]

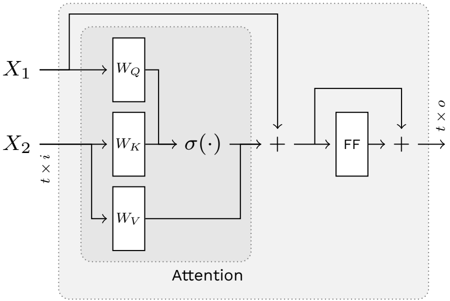
$$\text{Attention}(X_1, X_2) \;=\; \sigma\Big((X_1 W_Q)(X_2 W_K)^\top\Big) X_2 W_V$$



Set Transformers

Idea: Model track interactions through Self-Attention[2]

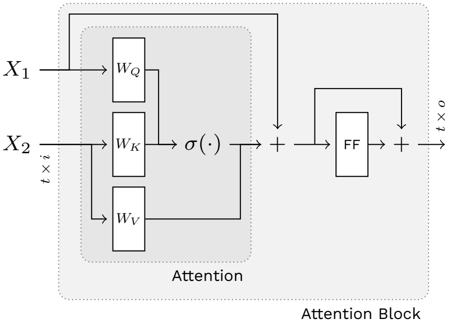
$$\text{Attention}(X_1, X_2) \;=\; \sigma\Big((X_1 W_Q)(X_2 W_K)^\top\Big) X_2 W_V$$



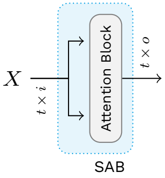
Set Transformers

Idea: Model track interactions through Self-Attention[2]

$$\text{Attention}(X_1, X_2) = \sigma\Big((X_1 W_Q)(X_2 W_K)^\top\Big) X_2 W_V$$



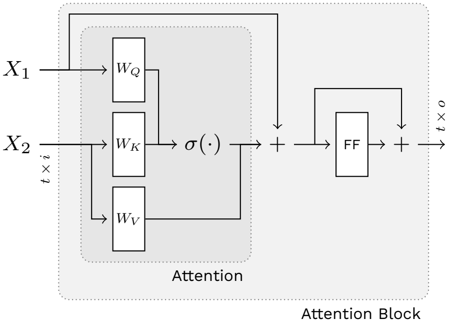
Self-Attention Block:



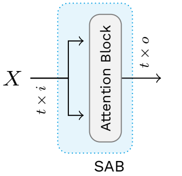
Set Transformers

Idea: Model track interactions through Self-Attention[2]

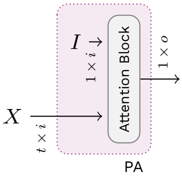
$$\text{Attention}(X_1, X_2) \;=\; \sigma\Big((X_1 W_Q)(X_2 W_K)^\top\Big) X_2 W_V$$



Self-Attention Block:



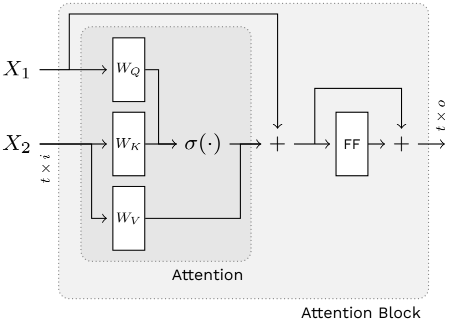
Pooling by Attention:



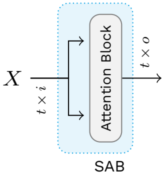
Set Transformers

Idea: Model track interactions through Self-Attention[2]

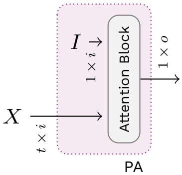
$$\text{Attention}(X_1, X_2) = \sigma\Big((X_1 W_Q)(X_2 W_K)^\top\Big) X_2 W_V$$



Self-Attention Block:



Pooling by Attention:



Set Encoder: SAB ∘ ... ∘ SAB ∘ PA

Set Transformers

Universal Approximation ✓

Track interactions ✓

Over-fitting to simulations ✗

Unsupervised Domain Adaptation

Problem: Simulations and real data do not match perfectly, i.e. $\mathbb{P}_{\mathcal{S}}(X,Y) \neq \mathbb{P}_{\mathcal{T}}(X,Y)$

[3] Ganin et al., "Domain-Adversarial Training of Neural Networks", 2016

Unsupervised Domain Adaptation

Problem: Simulations and real data do not match perfectly, i.e. $\mathbb{P}_{\mathcal{S}}(X, Y) \neq \mathbb{P}_{\mathcal{T}}(X, Y)$

- Assume Concept Shift: $\mathbb{P}_{\mathcal{S}}(X \mid Y) \neq \mathbb{P}_{\mathcal{T}}(X \mid Y)$ and $\mathbb{P}_{\mathcal{S}}(Y) = \mathbb{P}_{\mathcal{T}}(Y)$
- Employ Unlabeled Data: $\mathrm{D}_{\mathcal{T}} = \left\{ x \sim \mathbb{P}_{\mathcal{T}}(X) \right\}$

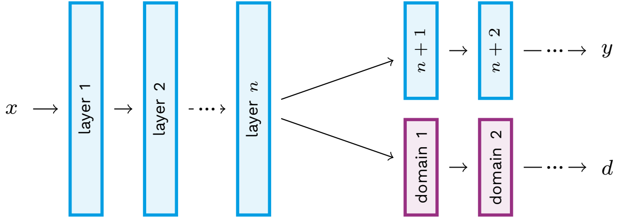
[3] Ganin et al., "Domain-Adversarial Training of Neural Networks", 2016

Unsupervised Domain Adaptation

Problem: Simulations and real data do not match perfectly, i.e. $\mathbb{P}_{\mathcal{S}}(X, Y) \neq \mathbb{P}_{\mathcal{T}}(X, Y)$

- Assume Concept Shift: $\mathbb{P}_{\mathcal{S}}(X \mid Y) \neq \mathbb{P}_{\mathcal{T}}(X \mid Y)$ and $\mathbb{P}_{\mathcal{S}}(Y) = \mathbb{P}_{\mathcal{T}}(Y)$
- Employ Unlabeled Data: $\mathrm{D}_{\mathcal{T}} = \left\{ x \sim \mathbb{P}_{\mathcal{T}}(X) \right\}$

Domain-Adversarial UDA:[3]



$$d = \begin{cases} 1 & \text{if } x \in \mathrm{D}_{\mathcal{T}} \\ 0 & \text{if } (x, y) \in \mathrm{D}_{\mathcal{S}} \end{cases}$$

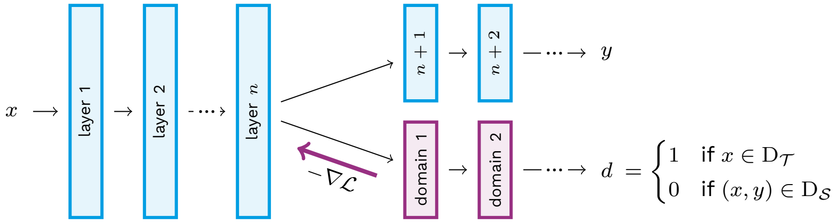
[3] Ganin et al., “Domain-Adversarial Training of Neural Networks”, 2016

Unsupervised Domain Adaptation

Problem: Simulations and real data do not match perfectly, i.e. $\mathbb{P}_{\mathcal{S}}(X, Y) \neq \mathbb{P}_{\mathcal{T}}(X, Y)$

- Assume Concept Shift: $\mathbb{P}_{\mathcal{S}}(X \mid Y) \neq \mathbb{P}_{\mathcal{T}}(X \mid Y)$ and $\mathbb{P}_{\mathcal{S}}(Y) = \mathbb{P}_{\mathcal{T}}(Y)$
- Employ Unlabeled Data: $\mathrm{D}_{\mathcal{T}} = \left\{x \sim \mathbb{P}_{\mathcal{T}}(X)\right\}$

Domain-Adversarial UDA:[3]



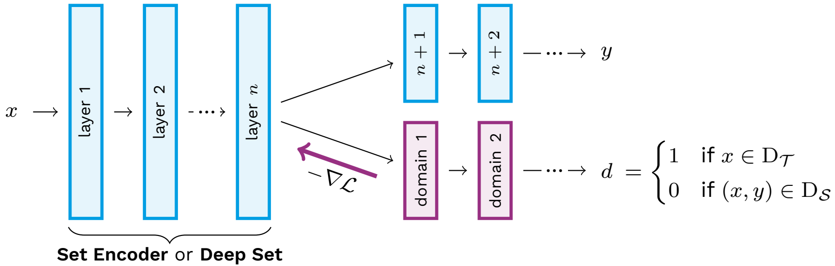
[3] Ganin et al., "Domain-Adversarial Training of Neural Networks", 2016

Unsupervised Domain Adaptation

Problem: Simulations and real data do not match perfectly, i.e. $\mathbb{P}_\mathcal{S}(X, Y) \neq \mathbb{P}_\mathcal{T}(X, Y)$

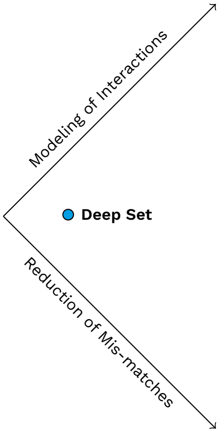
- Assume Concept Shift: $\mathbb{P}_\mathcal{S}(X \mid Y) \neq \mathbb{P}_\mathcal{T}(X \mid Y)$ and $\mathbb{P}_\mathcal{S}(Y) = \mathbb{P}_\mathcal{T}(Y)$
- Employ Unlabeled Data: $\mathrm{D}_\mathcal{T} = \left\{ x \sim \mathbb{P}_\mathcal{T}(X) \right\}$

Domain-Adversarial UDA:[3]

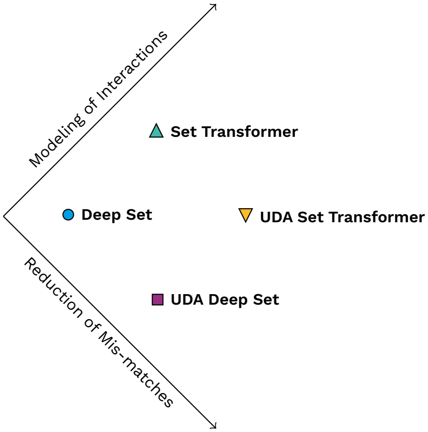


[3] Ganin et al., "Domain-Adversarial Training of Neural Networks", 2016

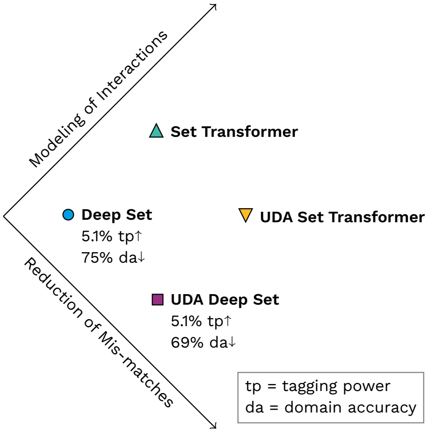
(Preliminary) Performance



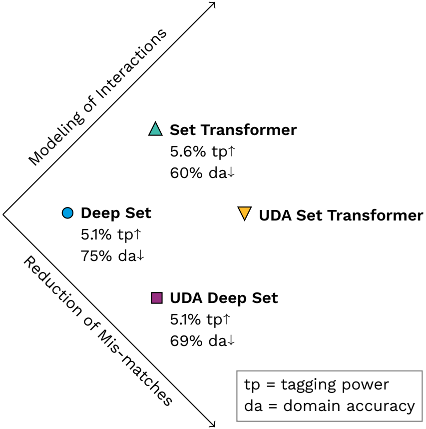
(Preliminary) Performance



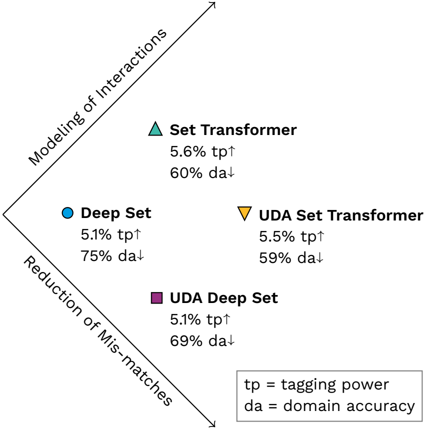
(Preliminary) Performance



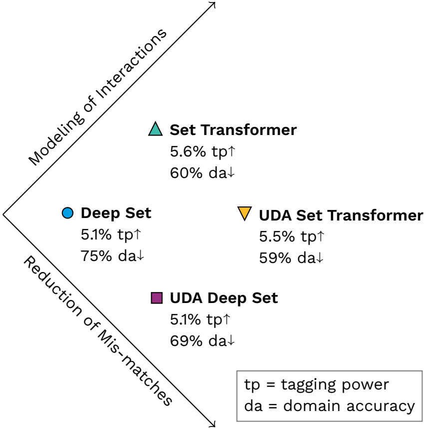
(Preliminary) Performance



(Preliminary) Performance



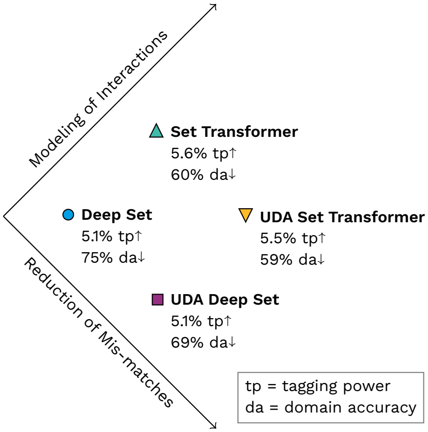
(Preliminary) Performance



(Preliminary) conclusions:

- Set Transformers increase tagging power and reduce mis-matches
- UDA does not improve tagging power (for now)

(Preliminary) Performance



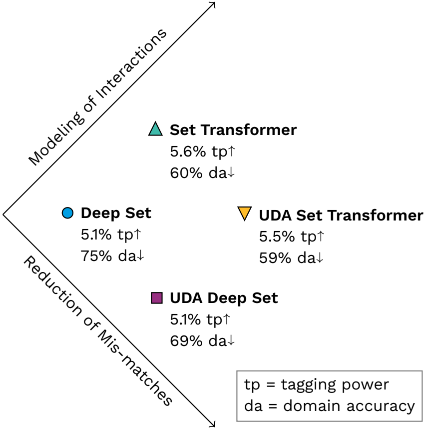
(Preliminary) conclusions:

- Set Transformers increase tagging power and reduce mis-matches
- UDA does not improve tagging power (for now)

Grains of salt:

- Difficulties in balancing the influence of UDA (currently: via multiple training phases)

(Preliminary) Performance



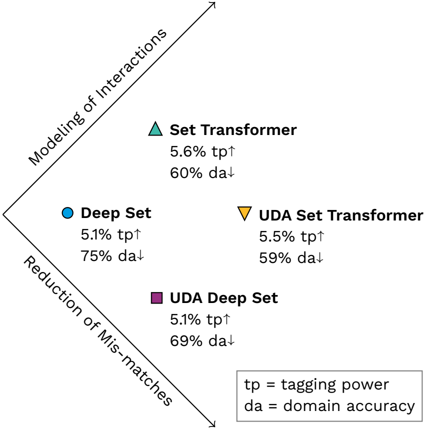
(Preliminary) conclusions:

- Set Transformers increase tagging power and reduce mis-matches
- UDA does not improve tagging power (for now)

Grains of salt:

- Difficulties in balancing the influence of UDA (currently: via multiple training phases)
- Unclear influence of factors besides mis-matches, like background and sweights

(Preliminary) Performance



(Preliminary) conclusions:

- Set Transformers increase tagging power and reduce mis-matches
- UDA does not improve tagging power (for now)

Grains of salt:

- Difficulties in balancing the influence of UDA (currently: via multiple training phases)
- Unclear influence of factors besides mis-matches, like background and sweights

Let’s discuss!