Towards Anytime Uncertainty Estimation in Early-Exit Neural Networks

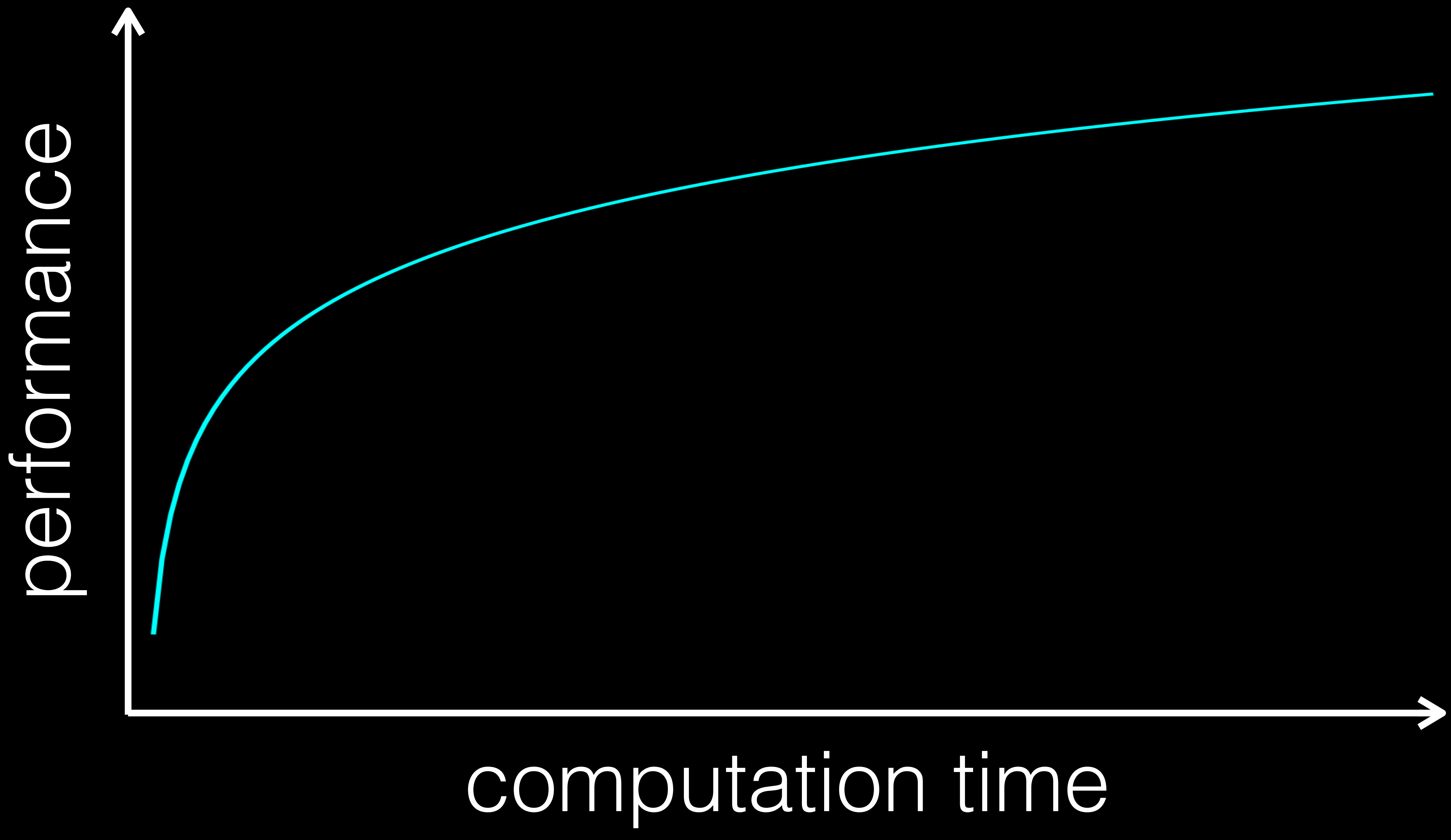
Eric Nalisnick

University of Amsterdam

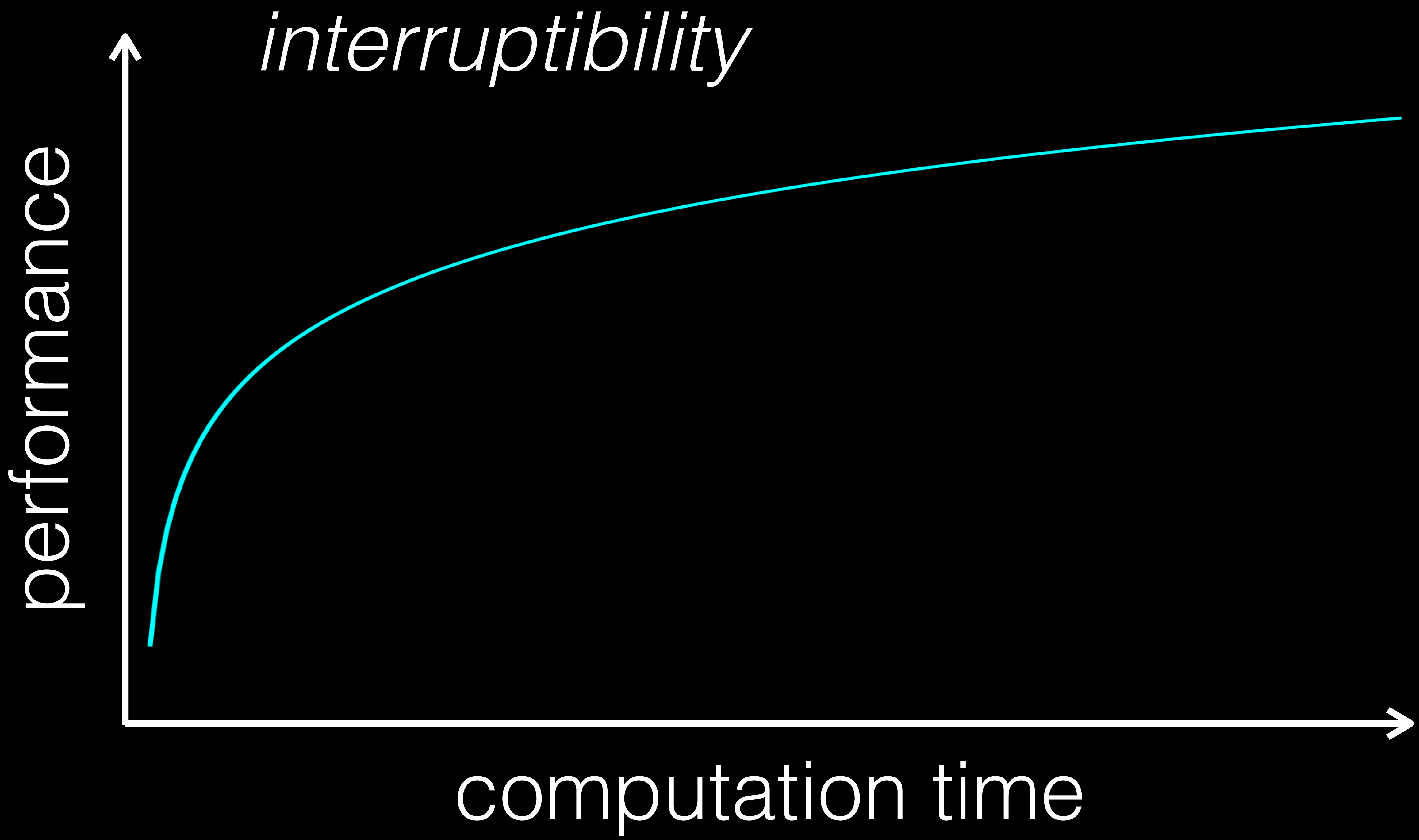
Amsterdam Machine Learning Lab


SELF DRIVING
MODE
40

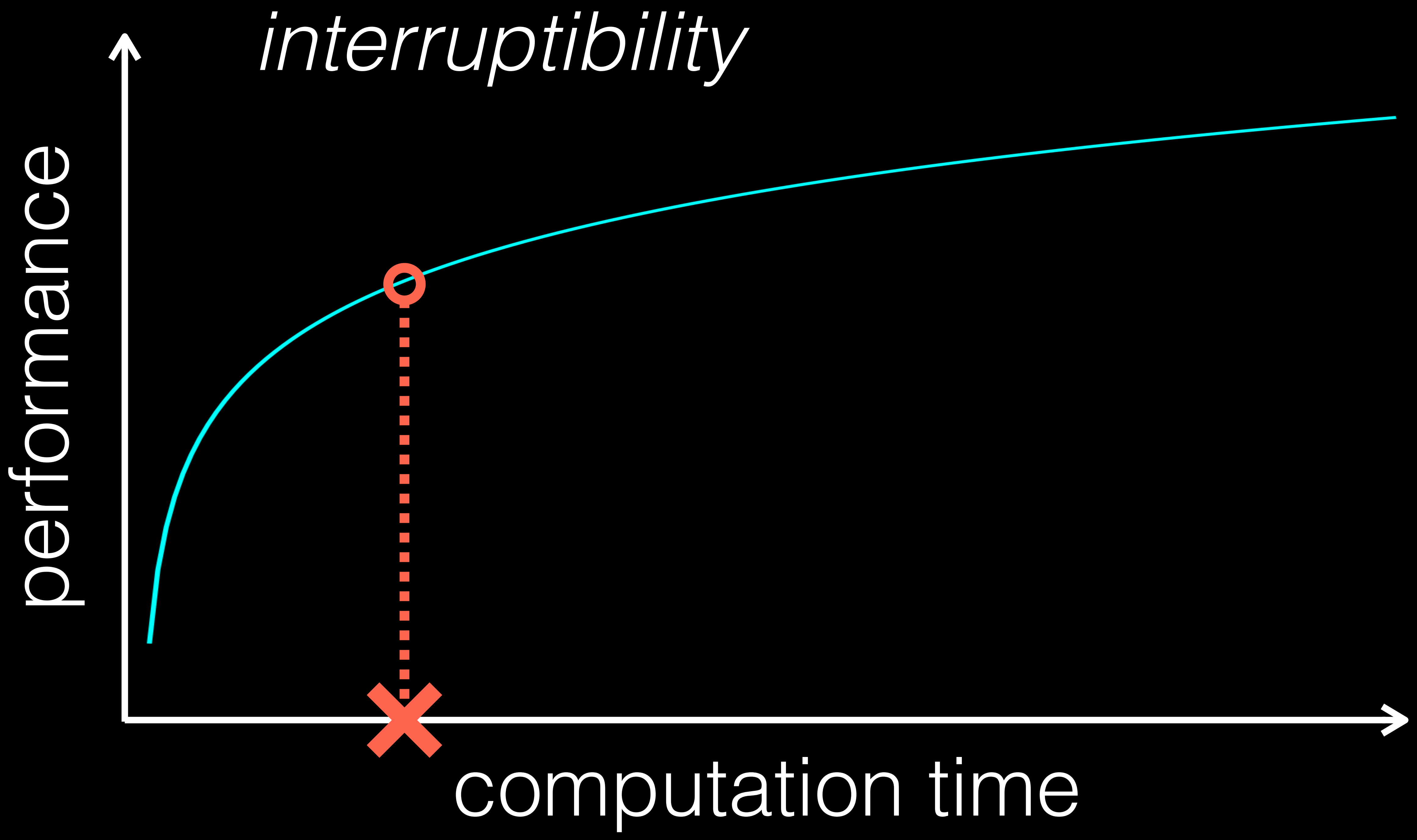
Anytime Models



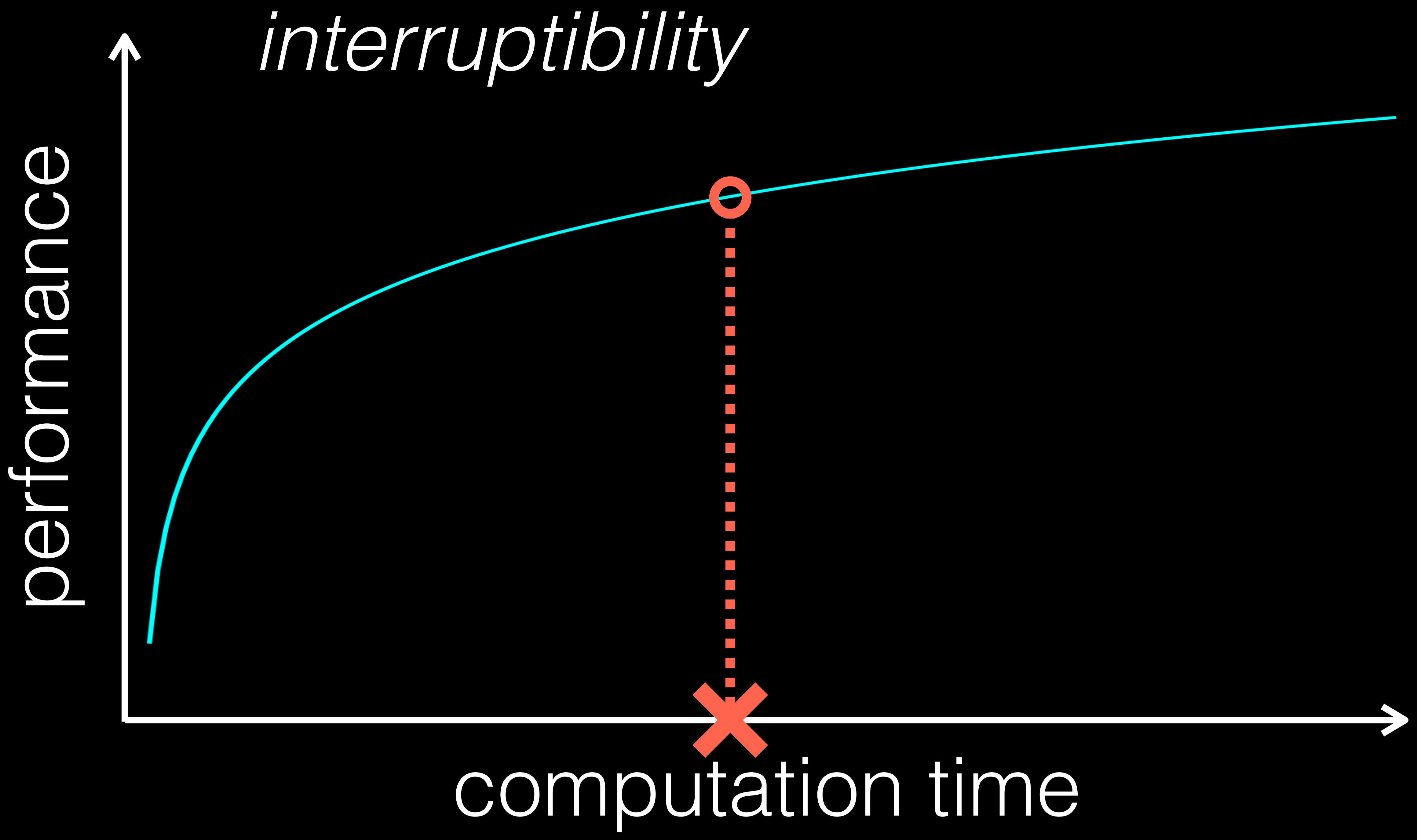
Anytime Models



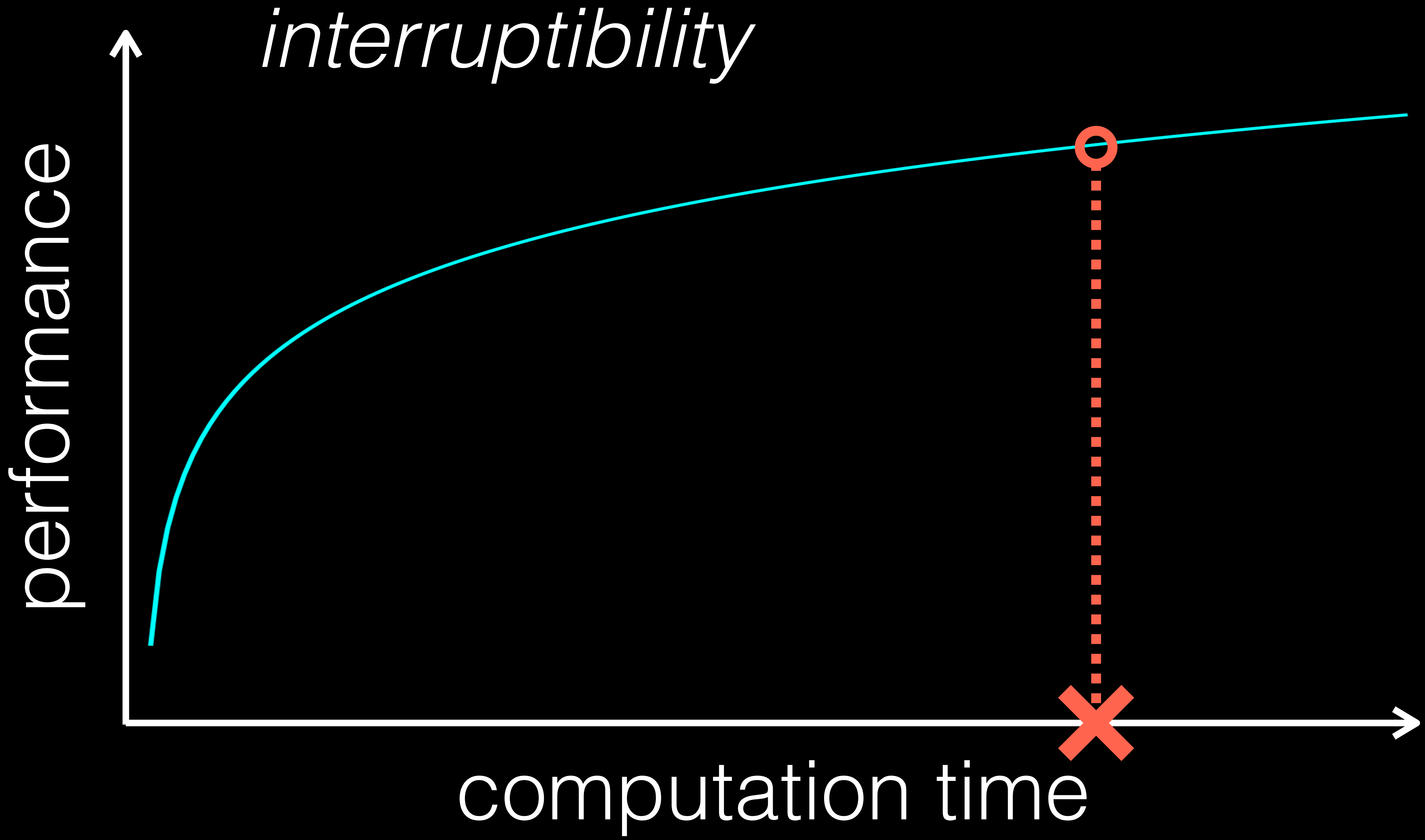
Anytime Models



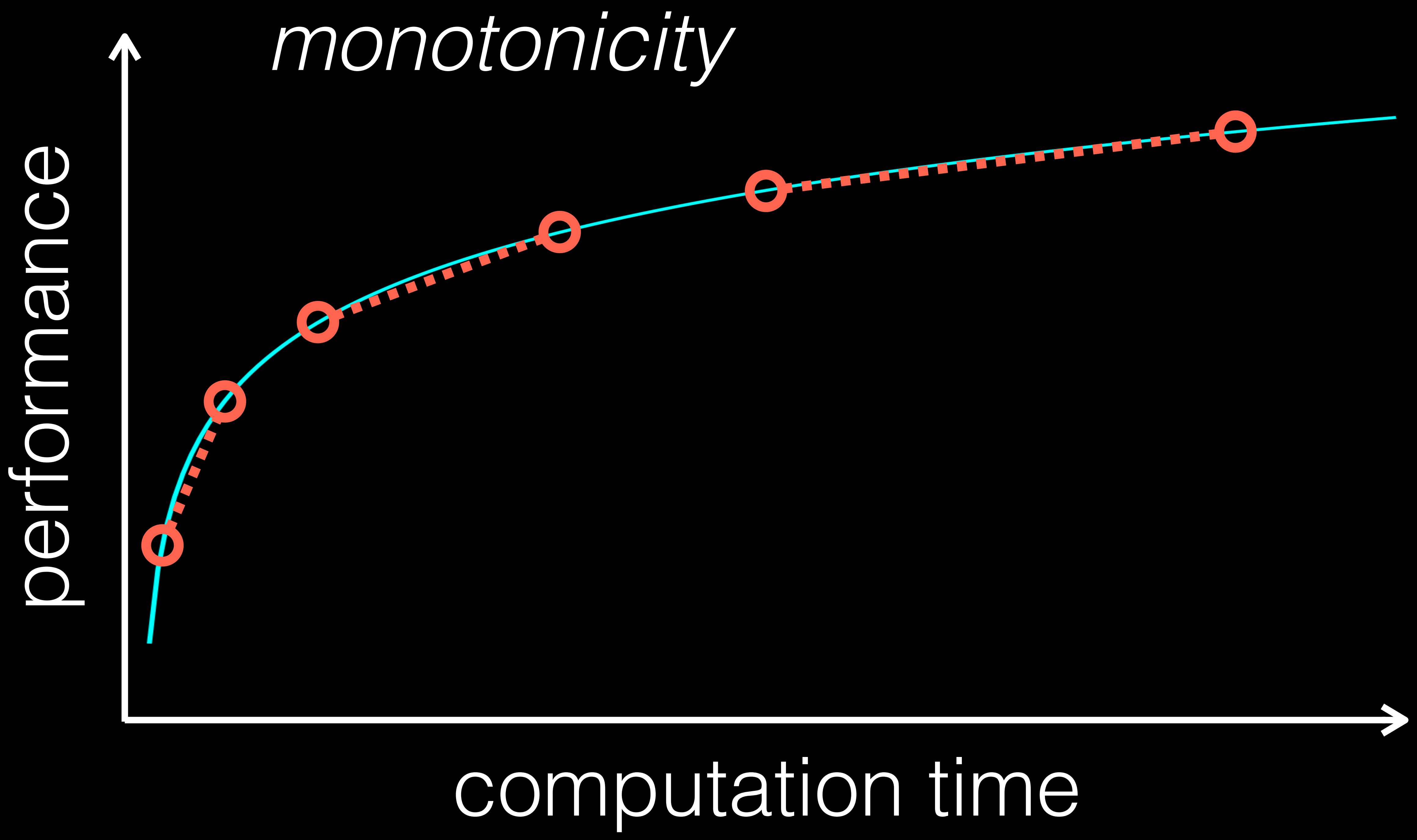
Anytime Models



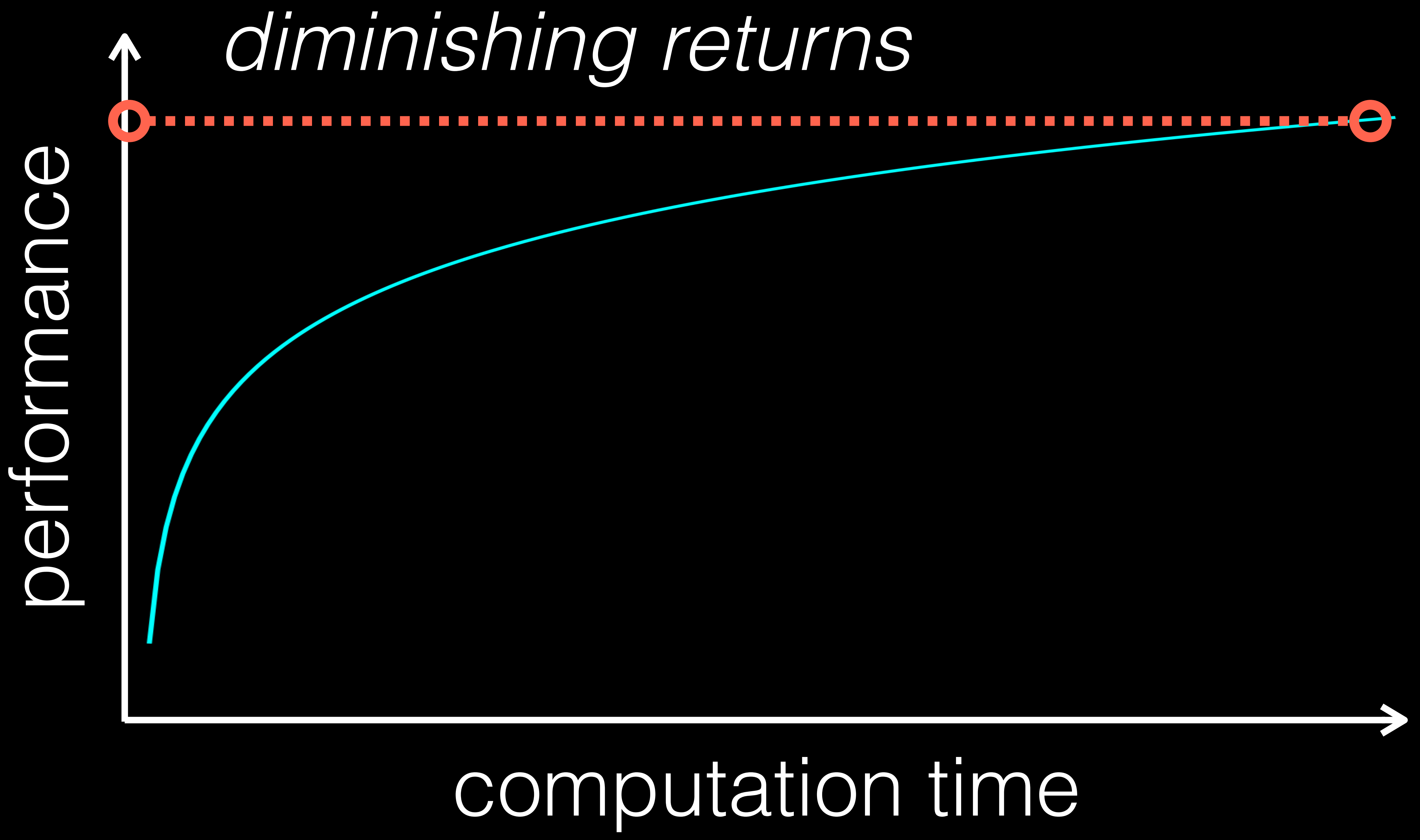
Anytime Models



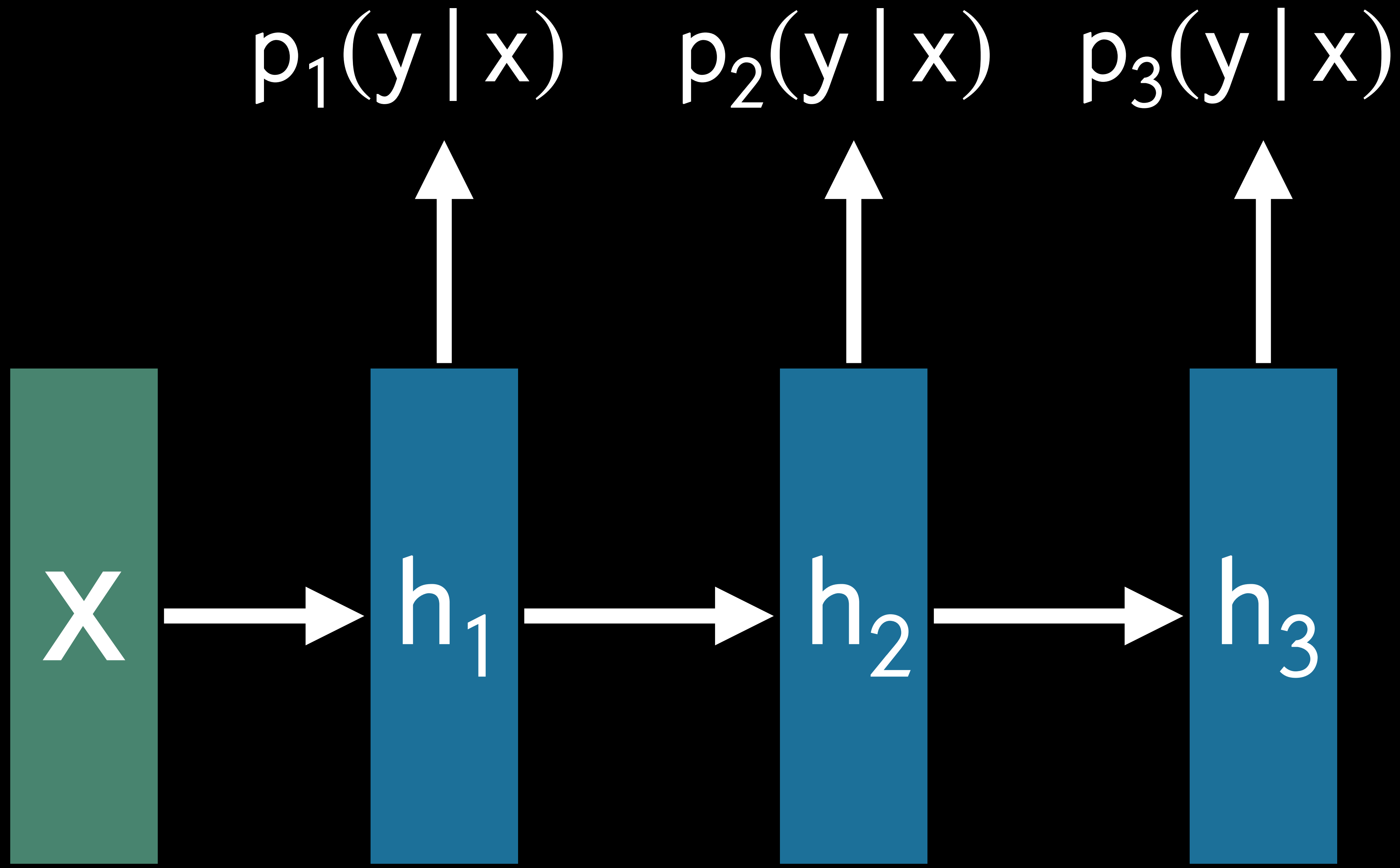
Anytime Models



Anytime Models

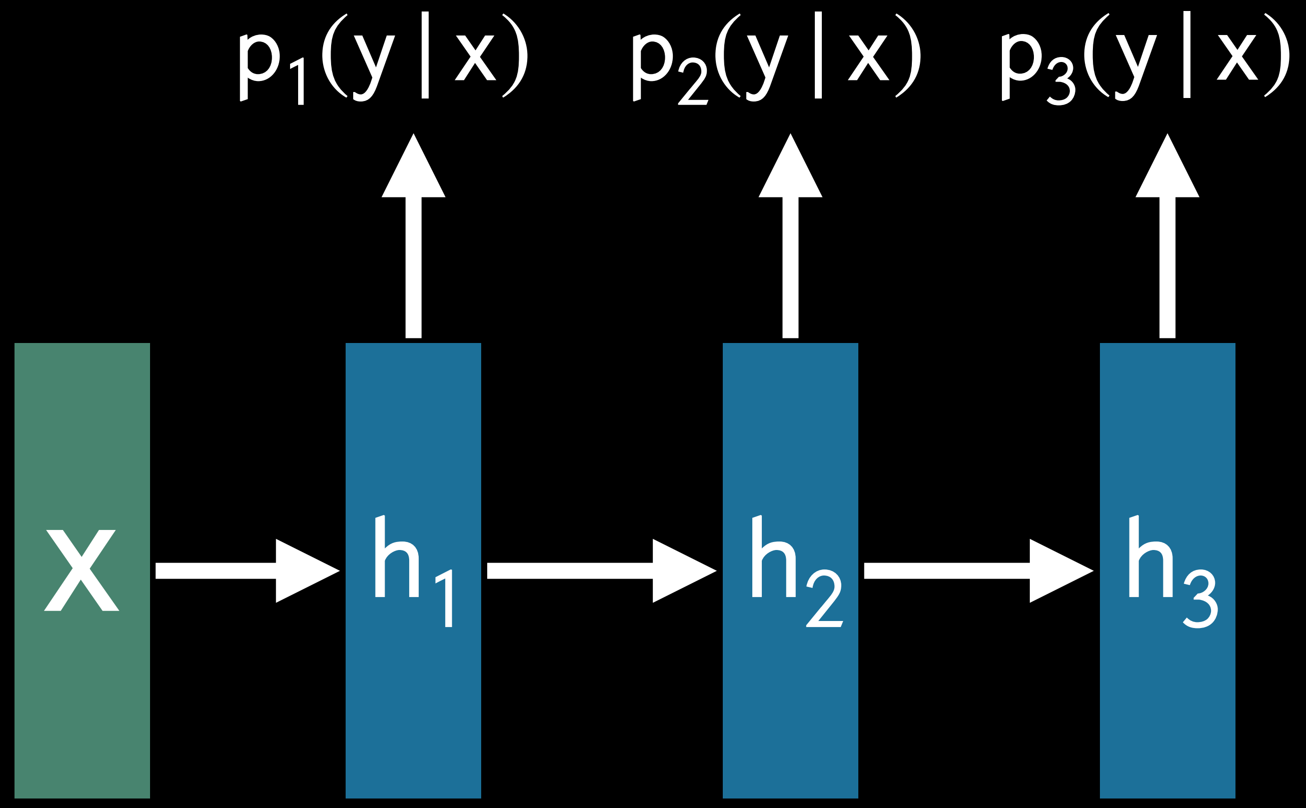


Early-Exit Neural Networks



Early-Exit Neural Networks

$$\ell\left(\theta_{1:E}\right) = -\sum_{e=1}^{E} \log p_e(y \mid x, \theta_{1:e})$$



Early-Exit Neural Networks

- ⊗ interruptibility?
- ⊗ monotonicity?
- ⊗ diminishing returns?

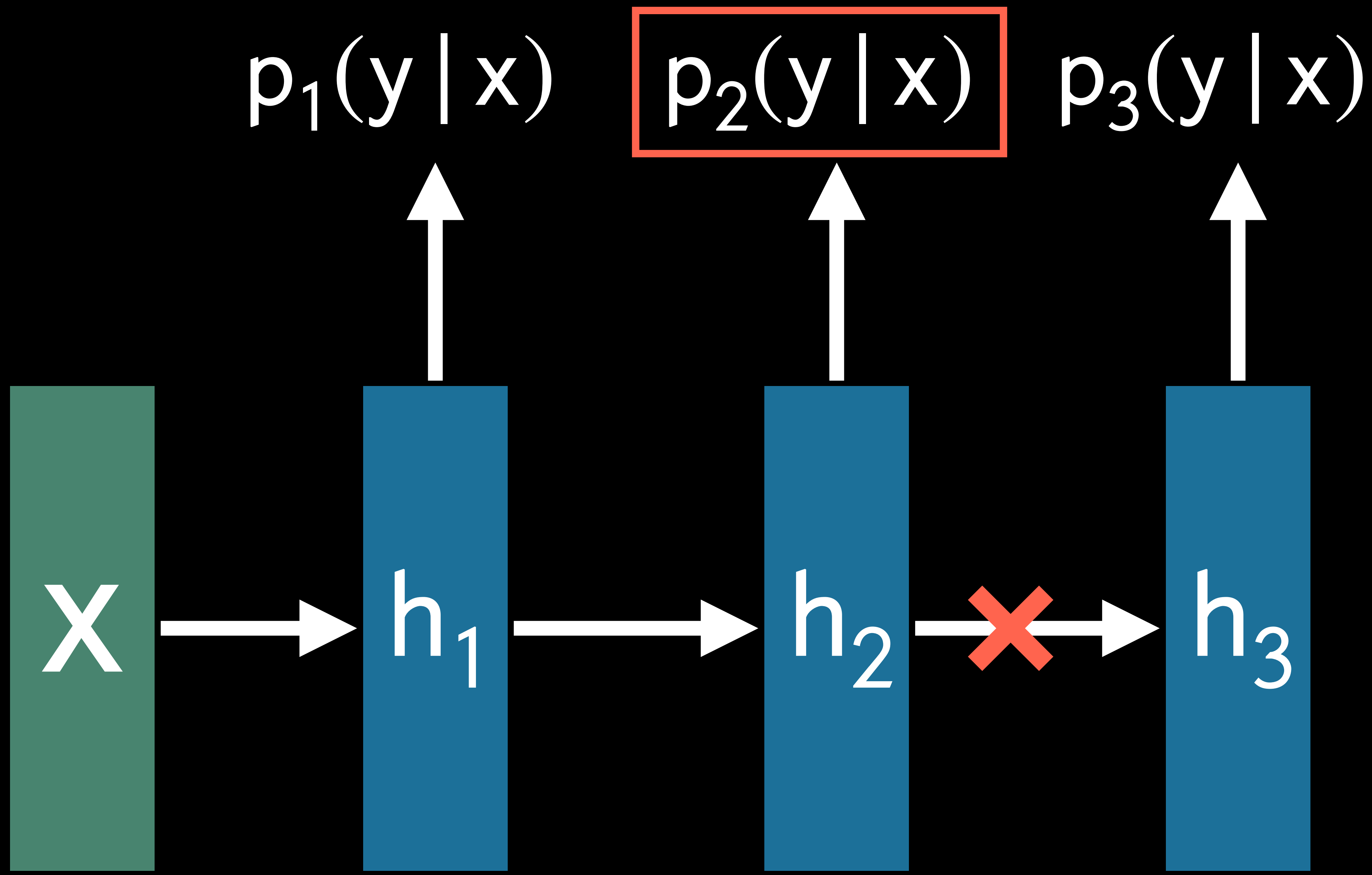
[Zilberstein, AI Magazine 1996]

Early-Exit Neural Networks

- ⊗ interruptibility?
- ⊗ monotonicity?
- ⊗ diminishing returns?

[Zilberstein, AI Magazine 1996]

Early-Exit Neural Networks



Early-Exit Neural Networks

- ⊗ interruptibility ✓
- ⊗ monotonicity?
- ⊗ diminishing returns?

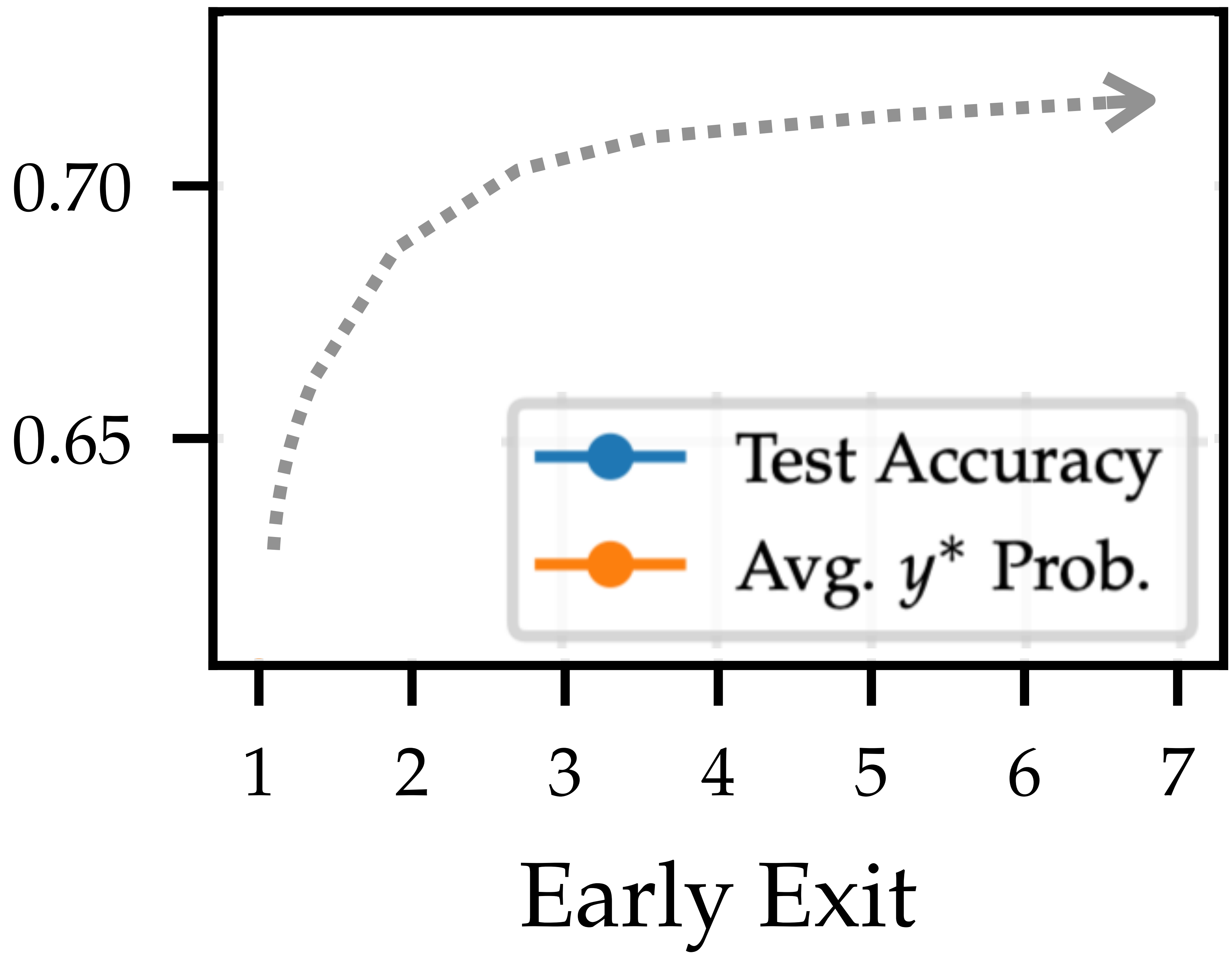
[Zilberstein, AI Magazine 1996]

Early-Exit Neural Networks

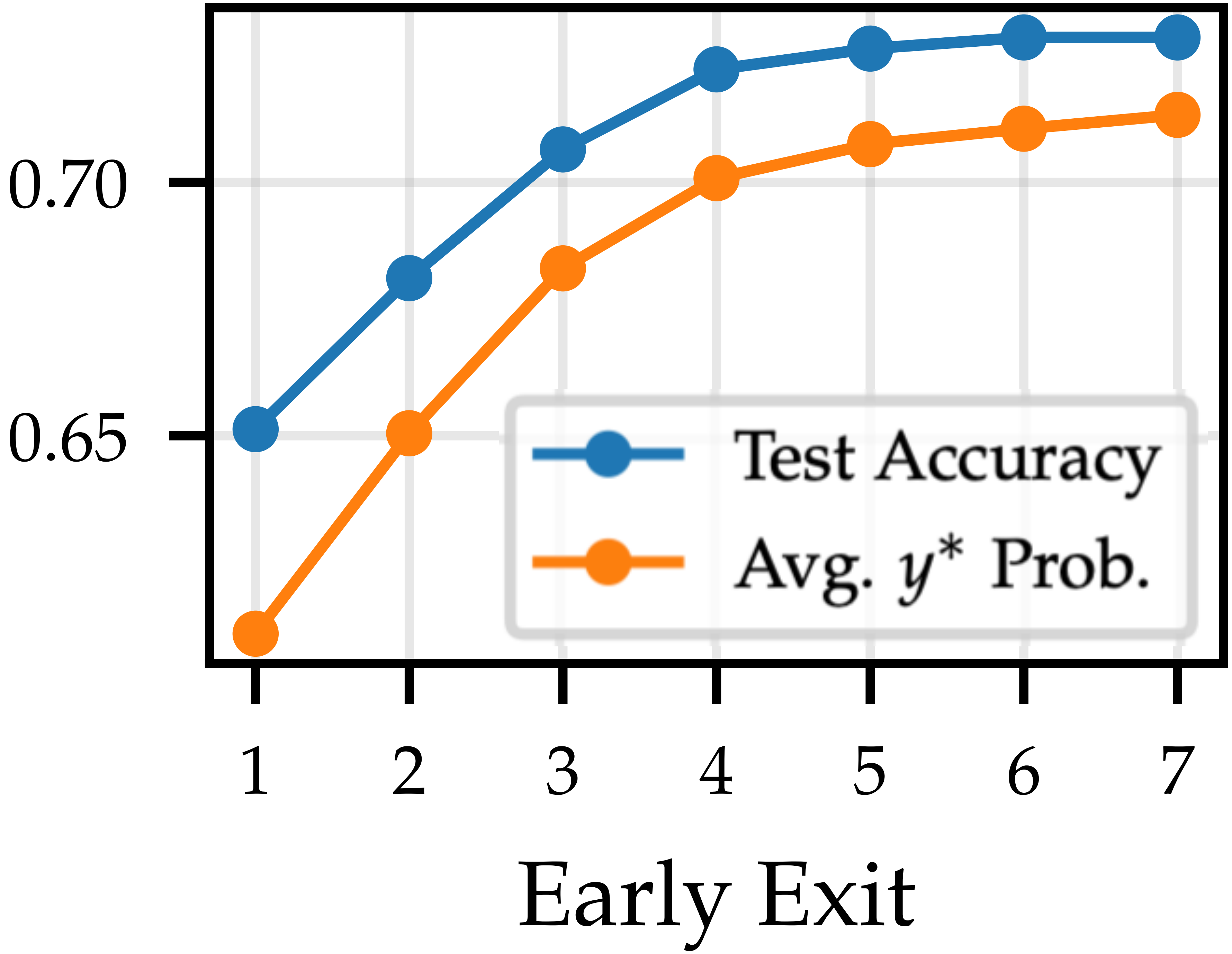
- ⊗ interruptibility ✓
- ⊗ monotonicity?
- ⊗ diminishing returns?

[Zilberstein, AI Magazine 1996]

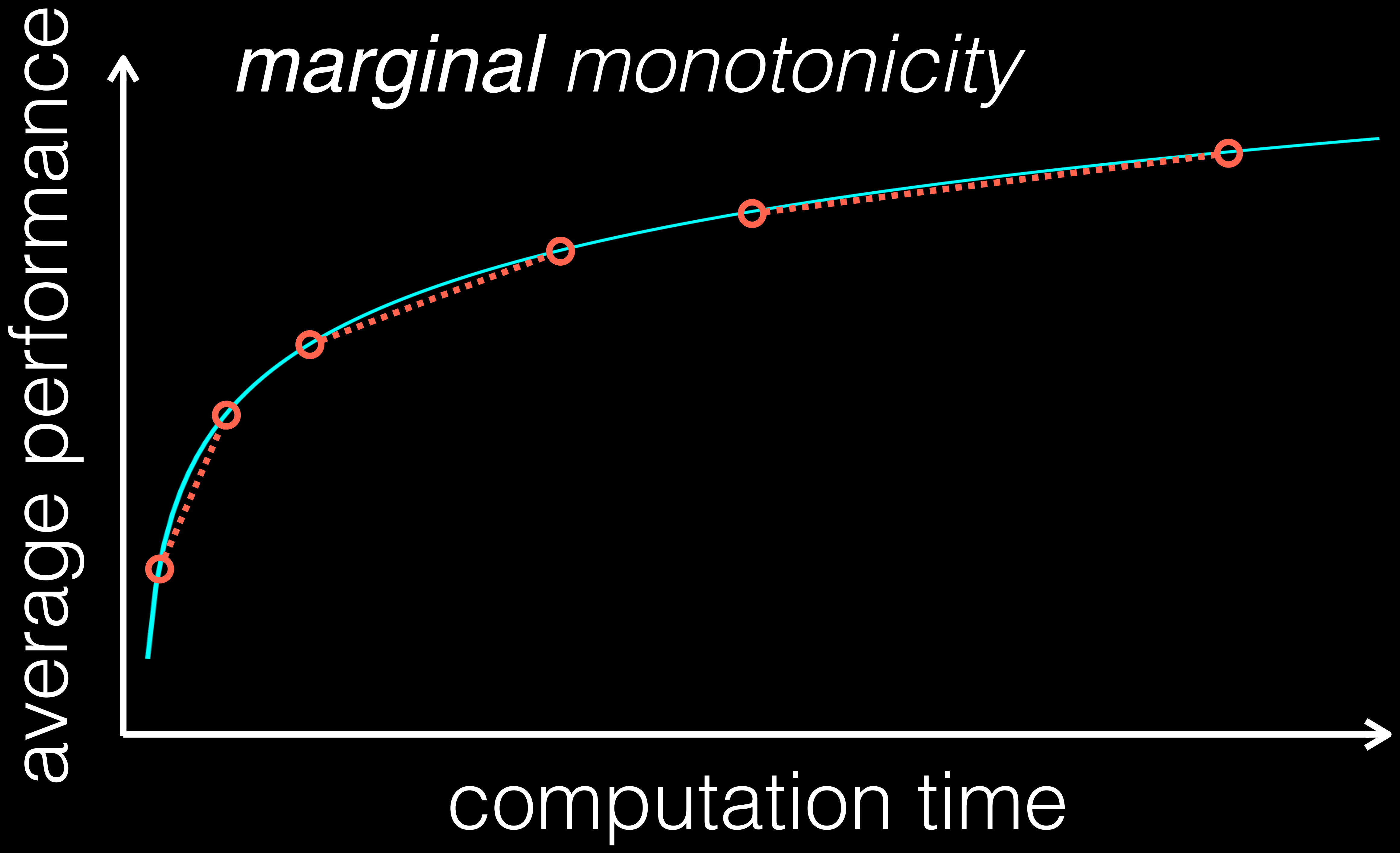
Multi-Scale Dense Net: CIFAR-100



Multi-Scale Dense Net: CIFAR-100



Anytime Models

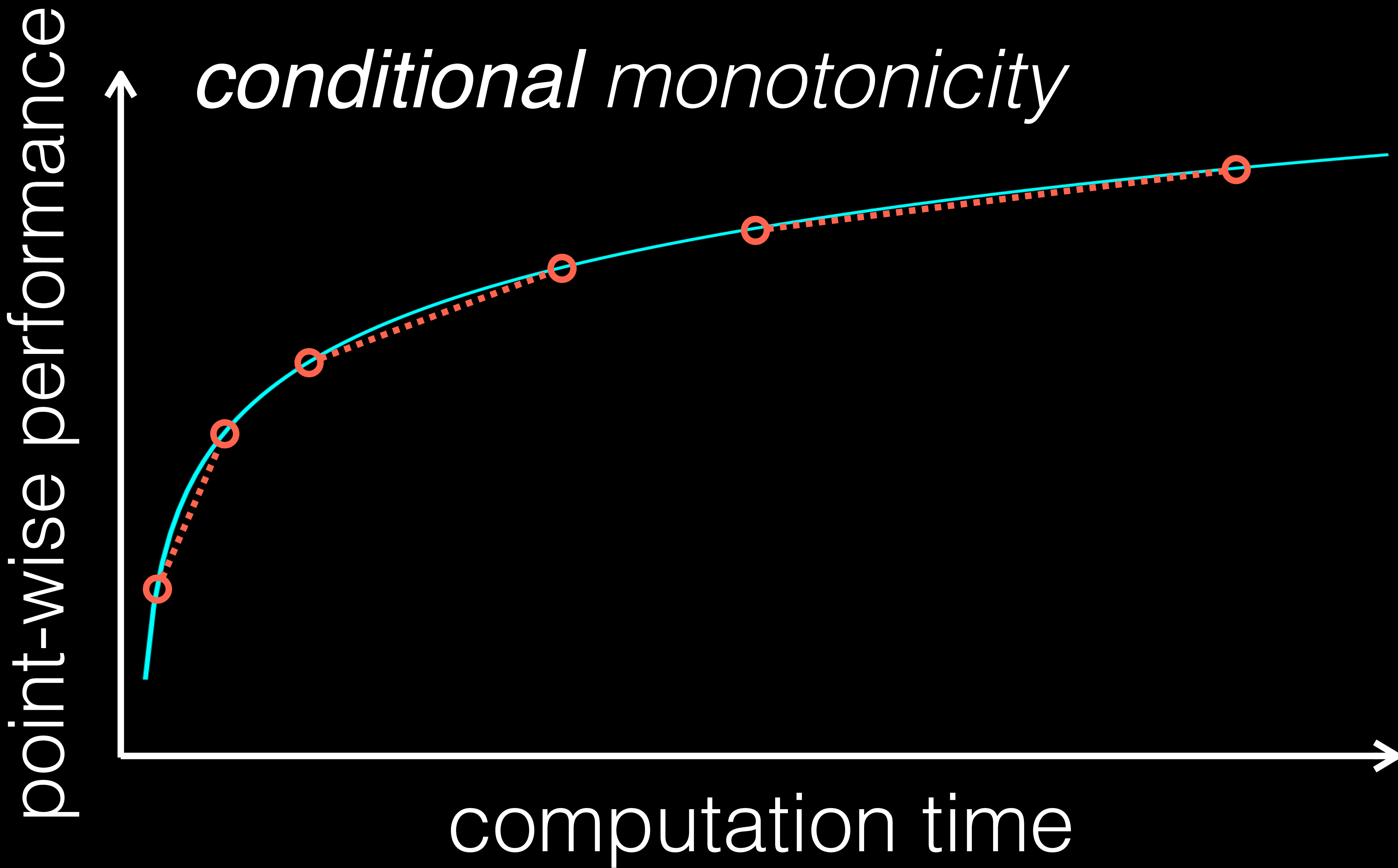


Early-Exit Neural Networks

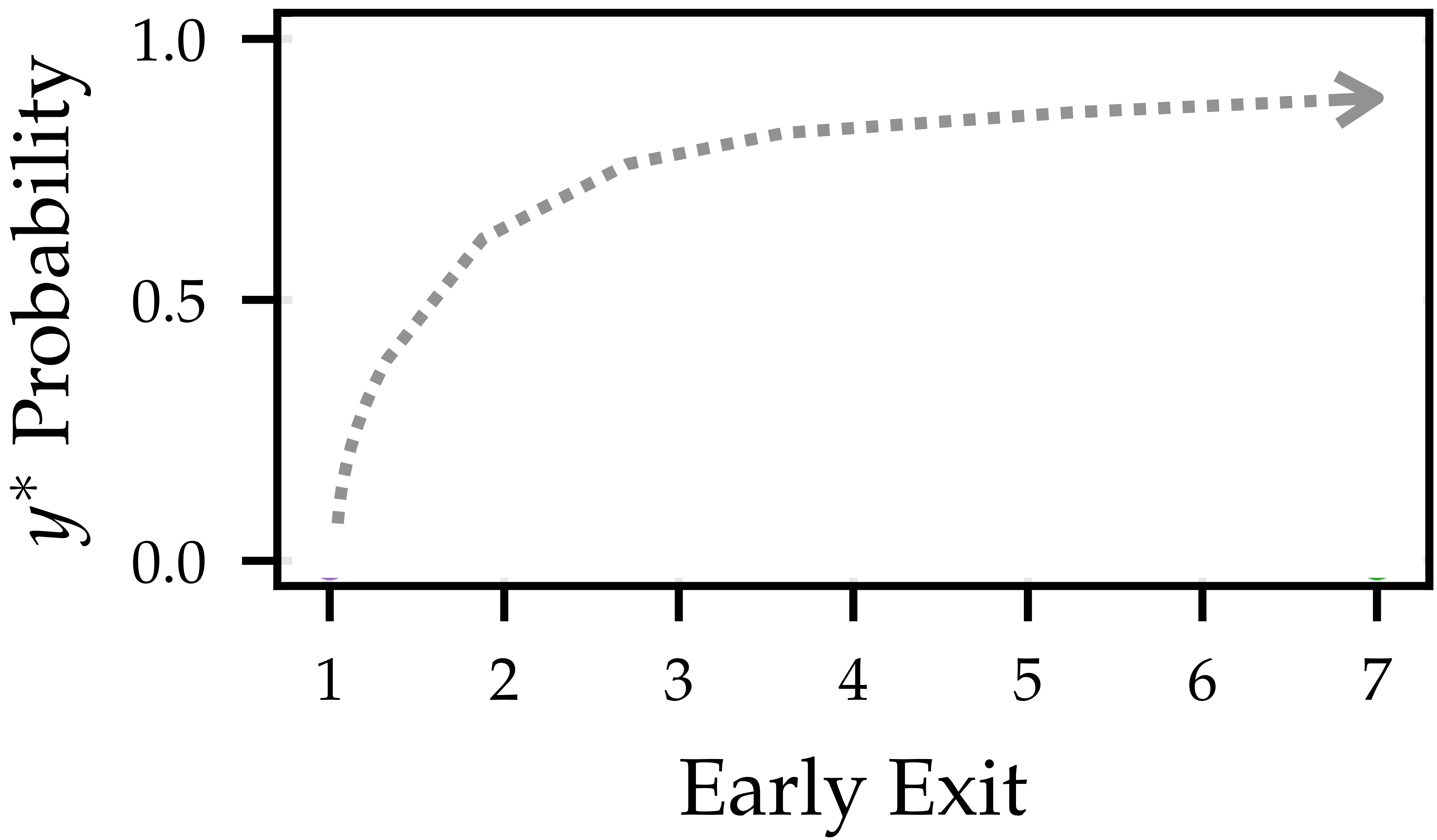
- ⊗ interruptibility ✓
- ⊗ monotonicity? ✓
- ⊗ diminishing returns?

[Zilberstein, AI Magazine 1996]

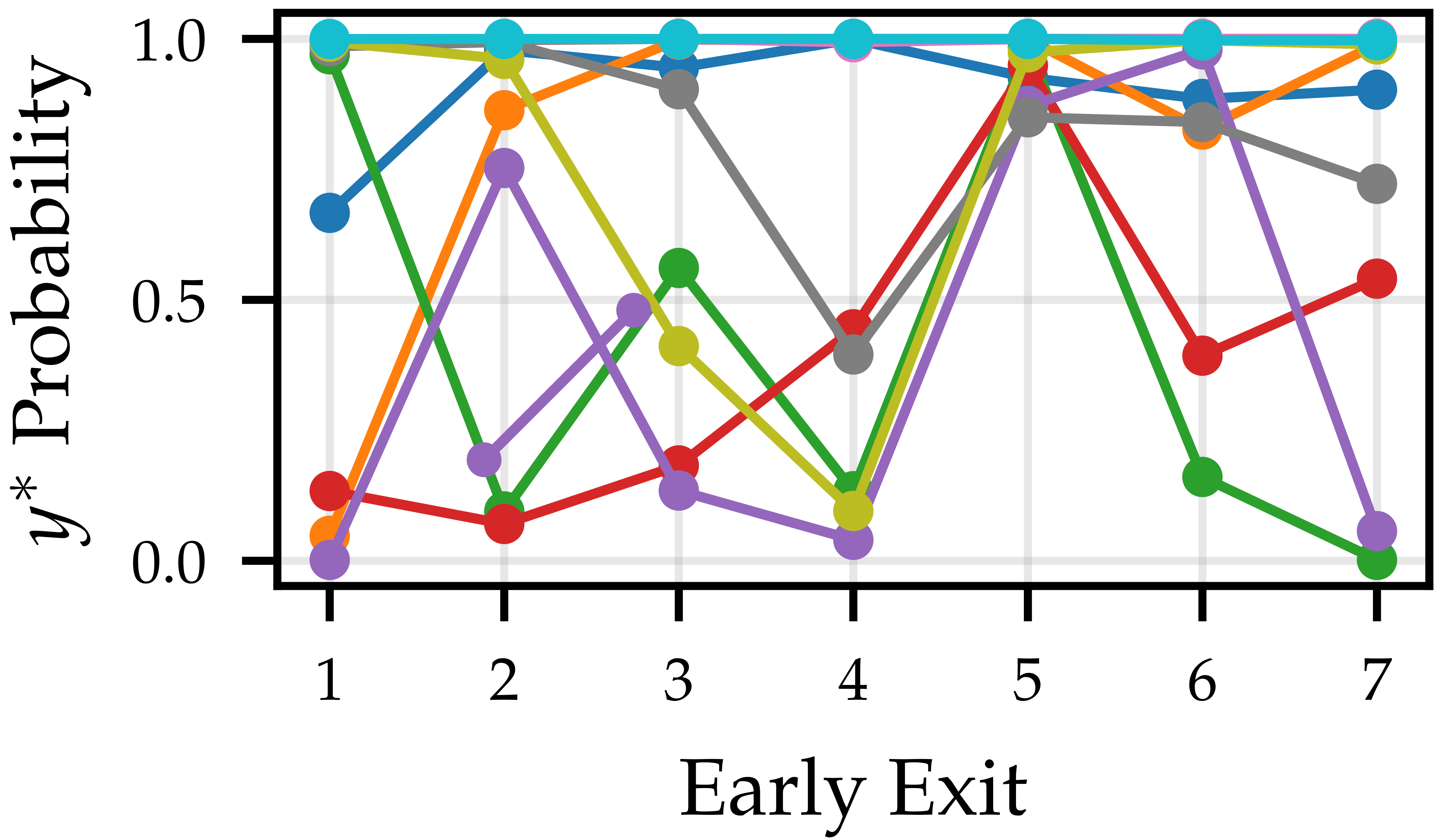
Anytime Models



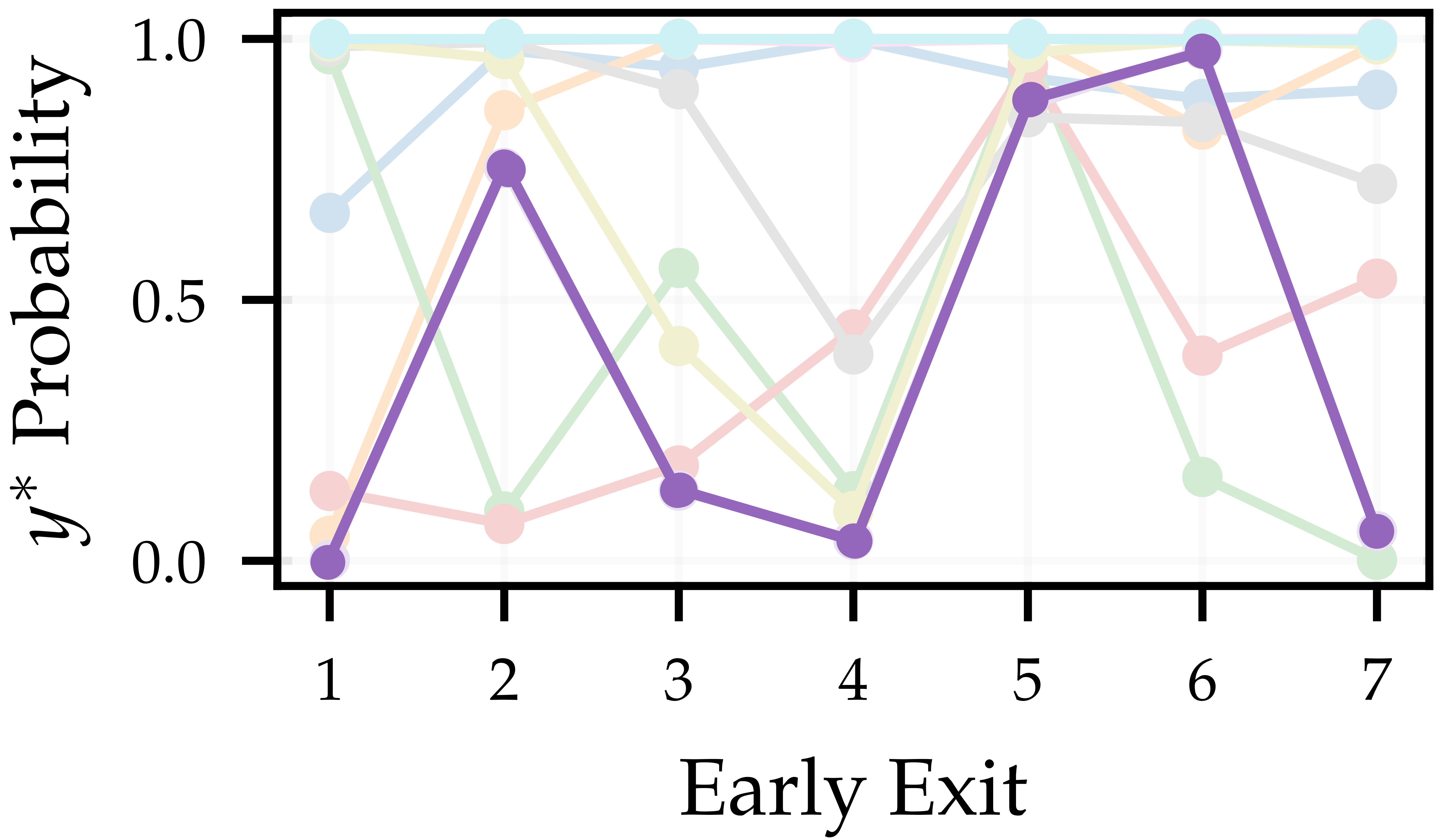
Multi-Scale Dense Net: CIFAR-100



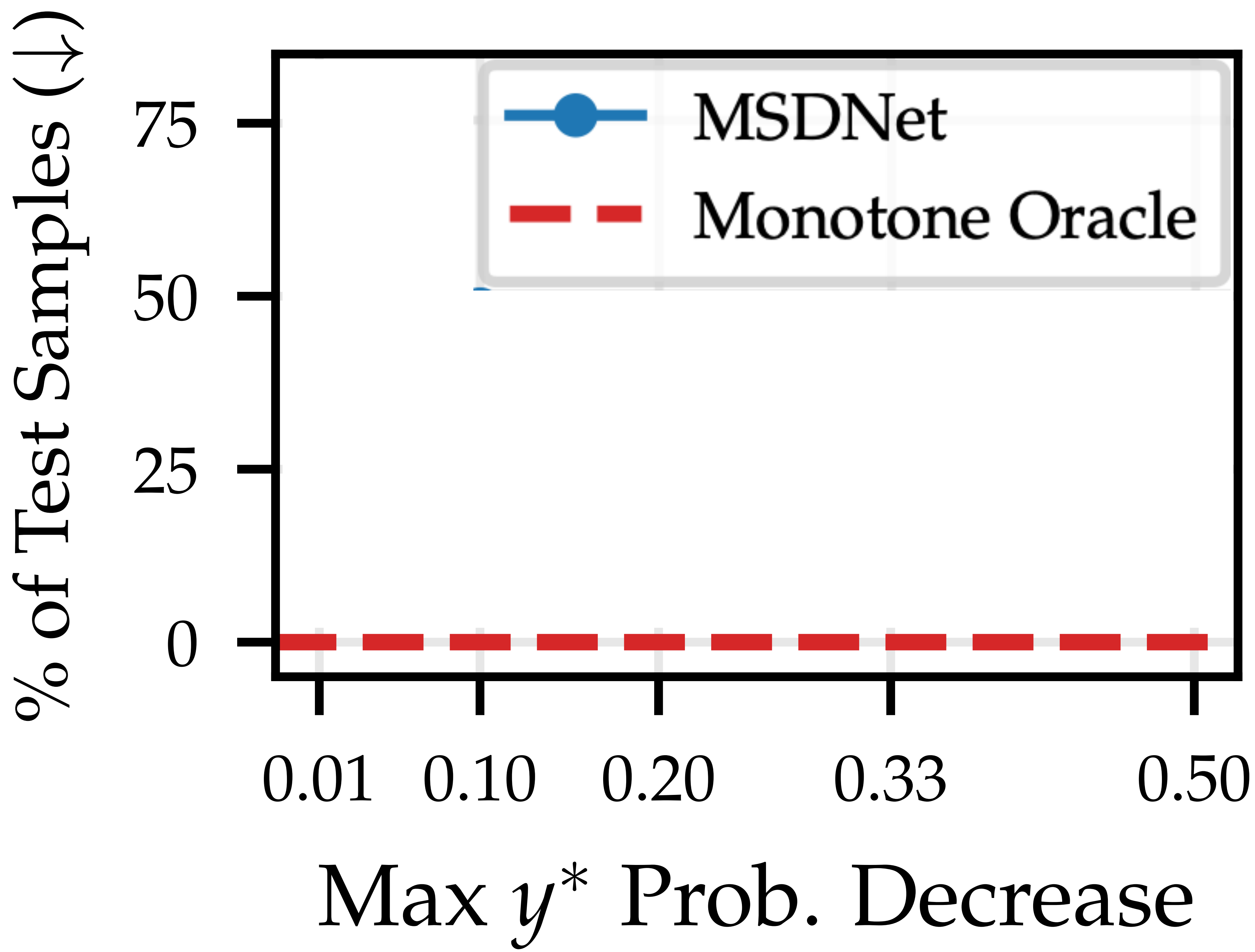
Multi-Scale Dense Net: CIFAR-100



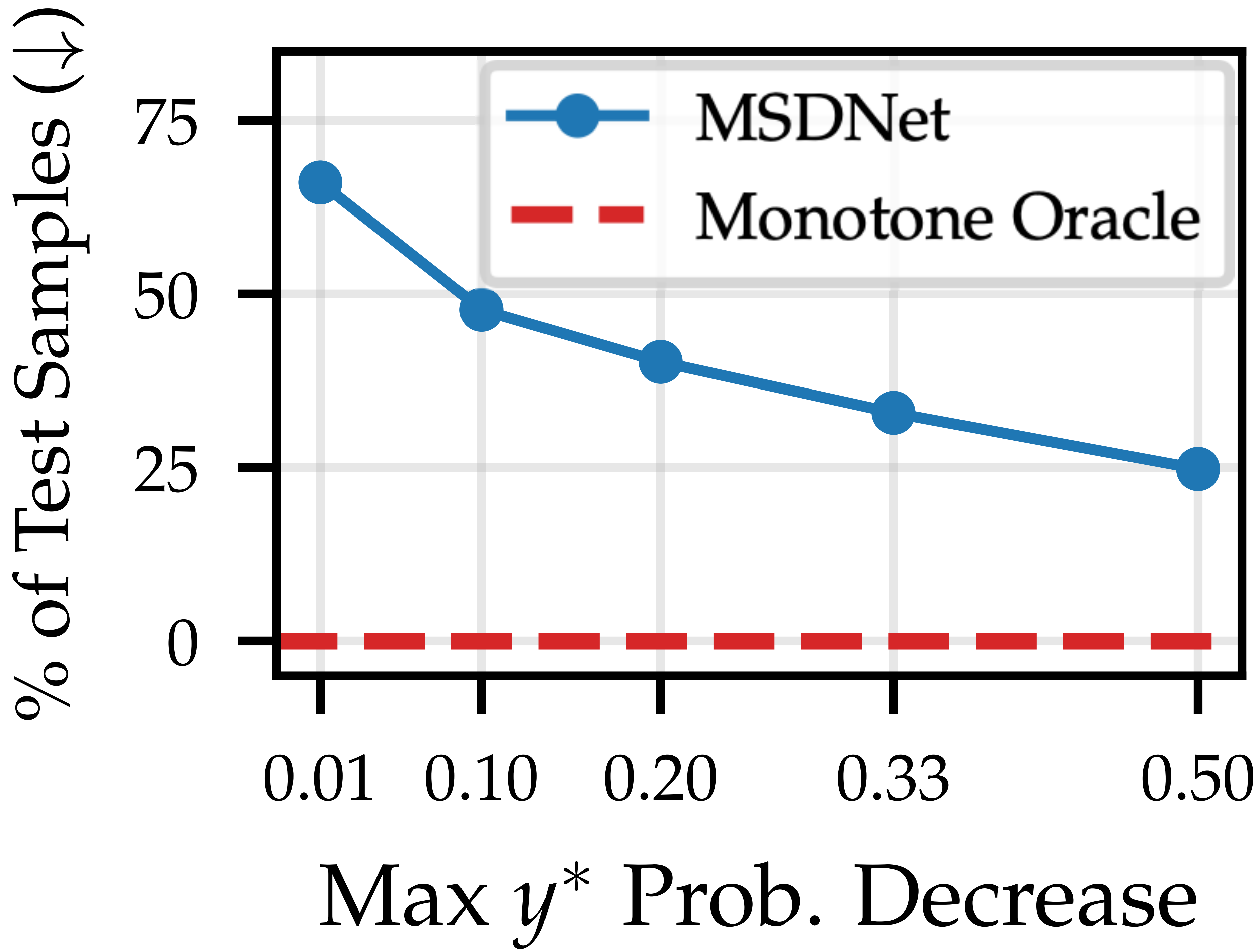
Multi-Scale Dense Net: CIFAR-100



Multi-Scale Dense Net: CIFAR-100



Multi-Scale Dense Net: CIFAR-100



Multi-Scale Dense Net: Overthinking

Overthinking: having the correct prediction but then switching to a wrong prediction.

[Kaya et al., ICML 2019]

$$\Delta = (\text{test error at final exit}) - (\text{test error if exited at correct prediction})$$

$$\Delta(\text{CIFAR} - 100) = \sim 14\,\%$$

$$\Delta(\text{ImageNet}) = \sim 9\,\%$$

Early-Exit Neural Networks

- ⊗ interruptibility
- ⊗ monotonicity
- ⊗ diminishing returns?

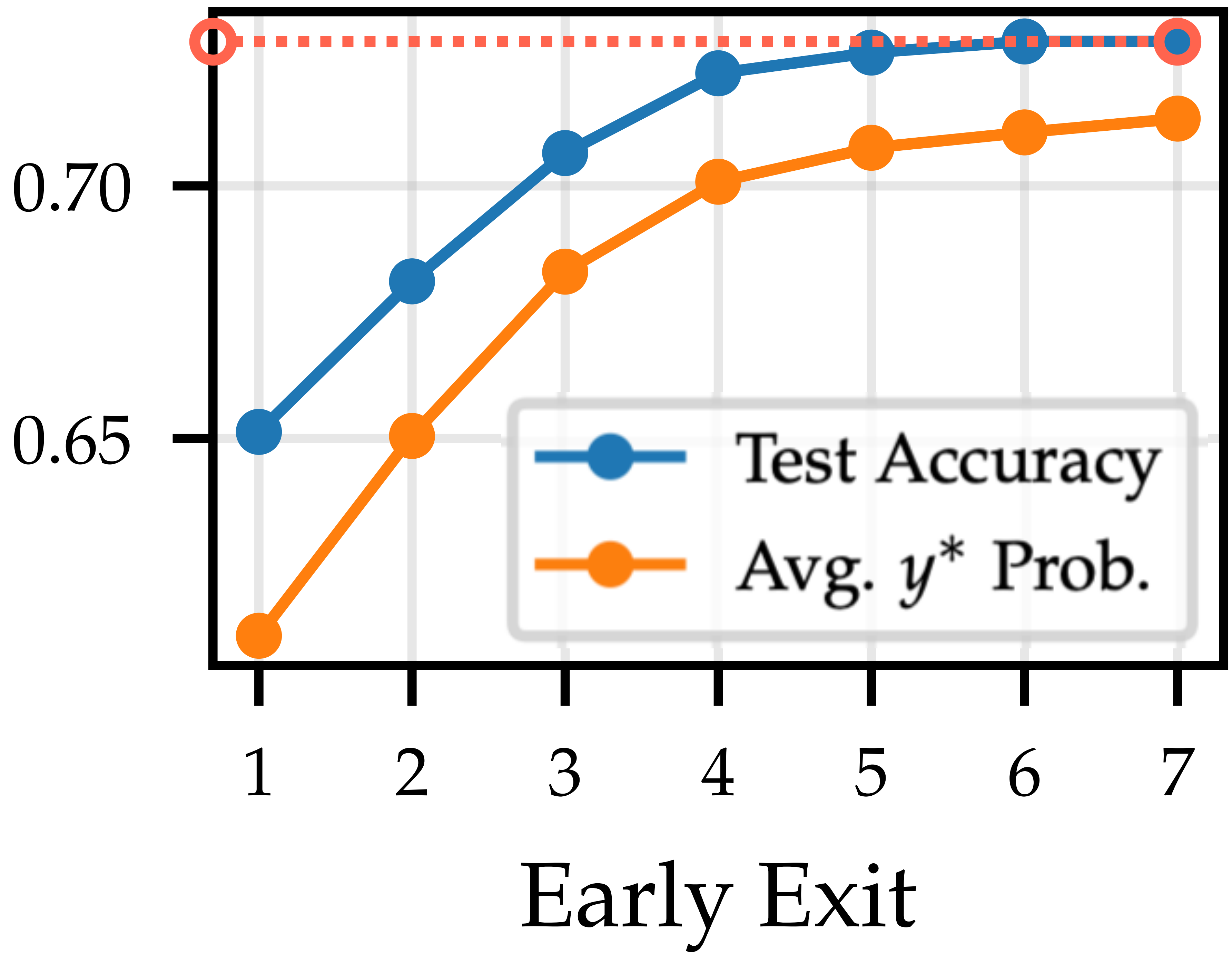
[Zilberstein, AI Magazine 1996]

Early-Exit Neural Networks

- ⊗ interruptibility
- ⊗ monotonicity
- ⊗ diminishing returns?

[Zilberstein, AI Magazine 1996]

Multi-Scale Dense Net: CIFAR-100



Early-Exit Neural Networks

- ⊗ interruptibility ✓
- ⊗ monotonicity ✓
- ⊗ diminishing returns ✓

[Zilberstein, AI Magazine 1996]

Early-Exit Neural Networks

- ⊗ interruptibility ✓
- ⊗ monotonicity ✓
- ⊗ diminishing returns ✓

only marginally

[Zilberstein, AI Magazine 1996]

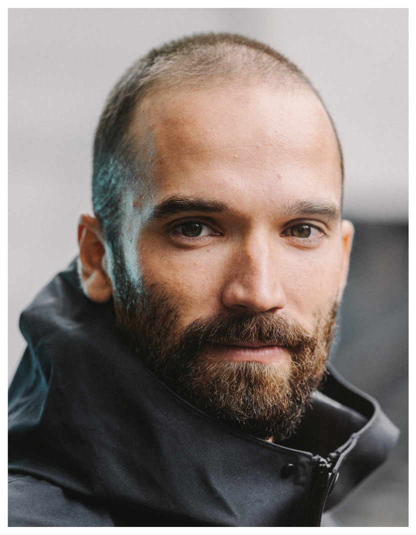
Early-Exit Neural Networks

- ⊗ interruptibility ✓
- ⊗ monotonicity ✓
- ⊗ diminishing returns ✓

[Zilberstein, AI Magazine 1996]

A simple, post-hoc method for encouraging conditional monotonicity

NEURAL INFORMATION PROCESSING SYSTEMS
2023

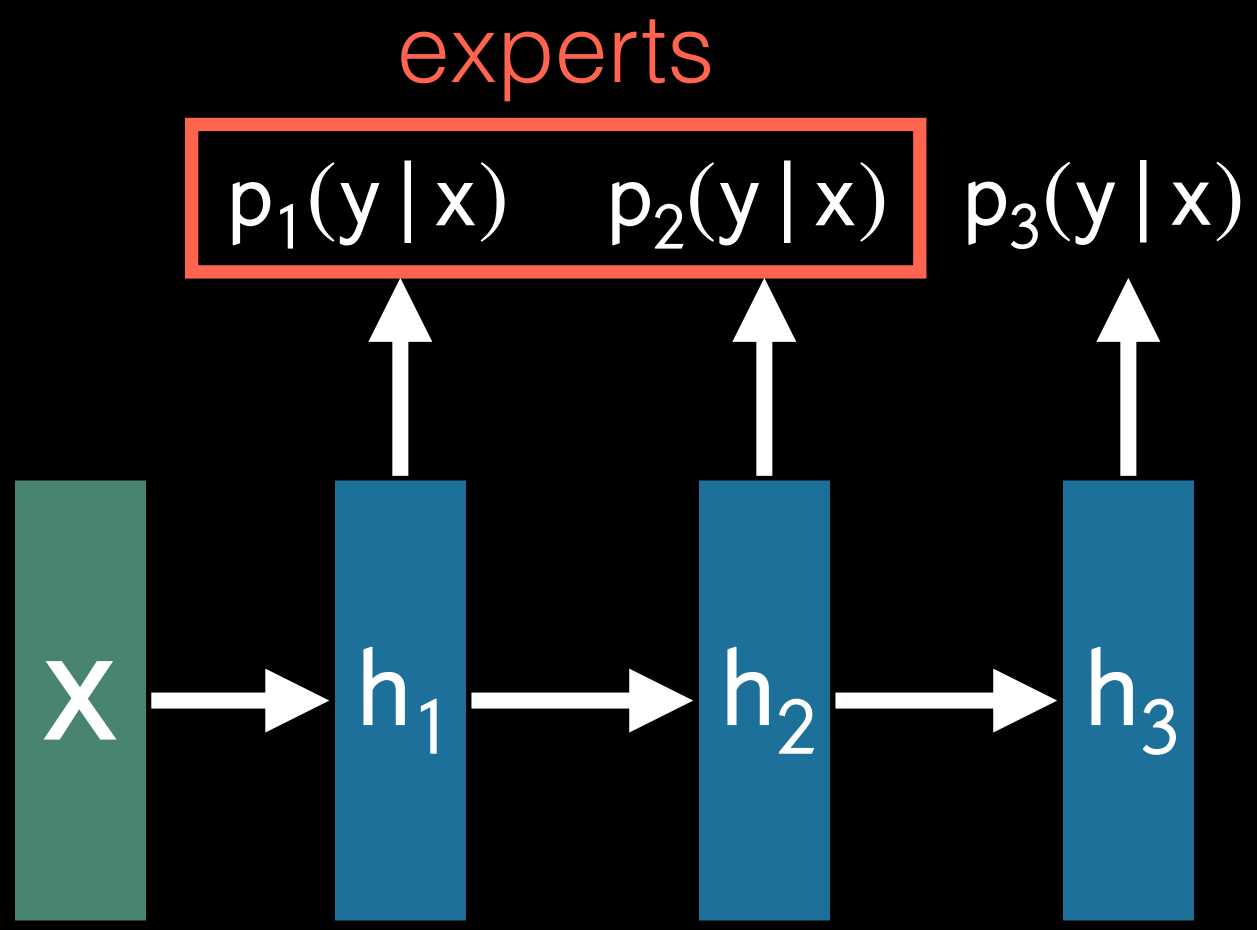


Metod Jazbec

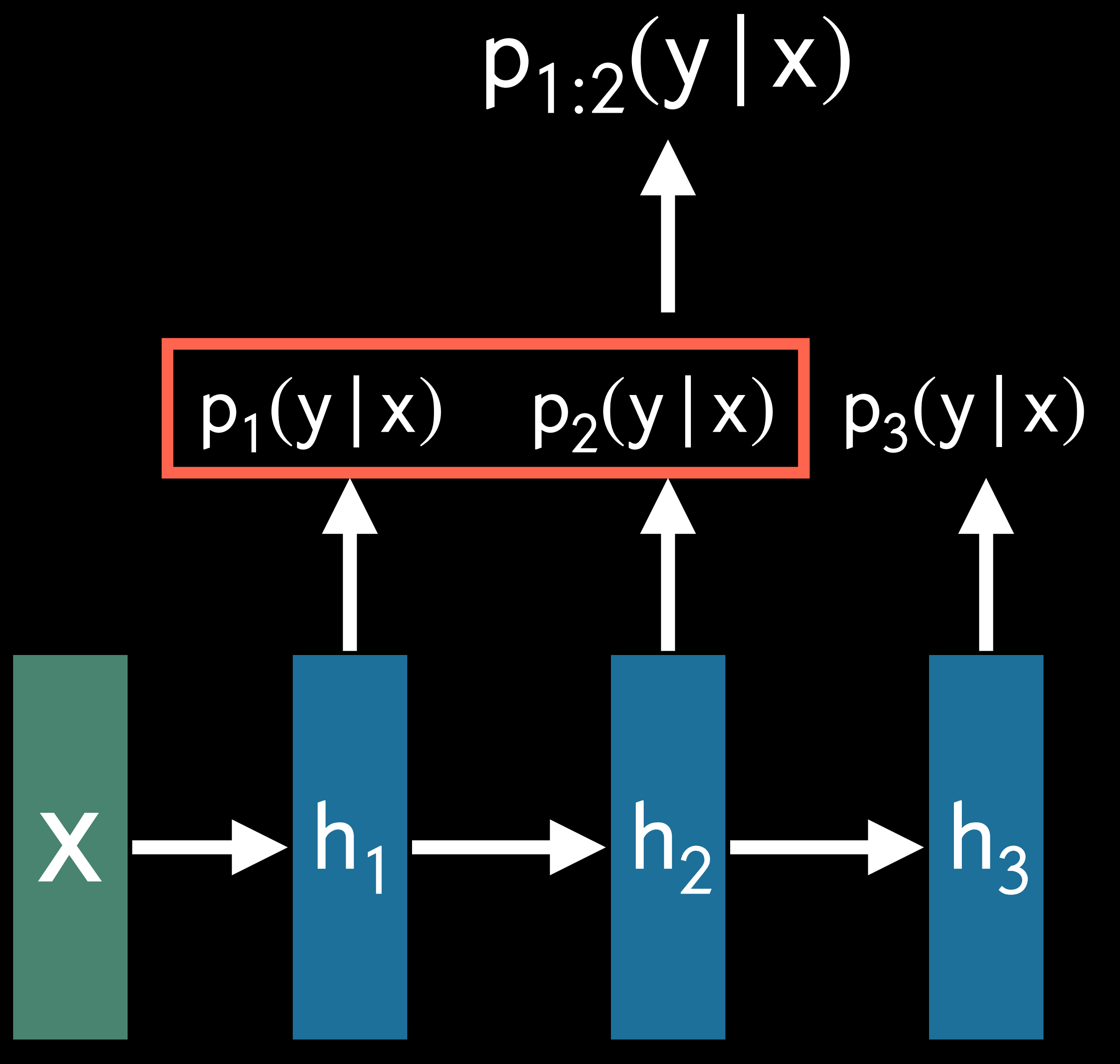
James U. Allingham

Dan Zhang

Idea: combine the early-exits via a product of experts

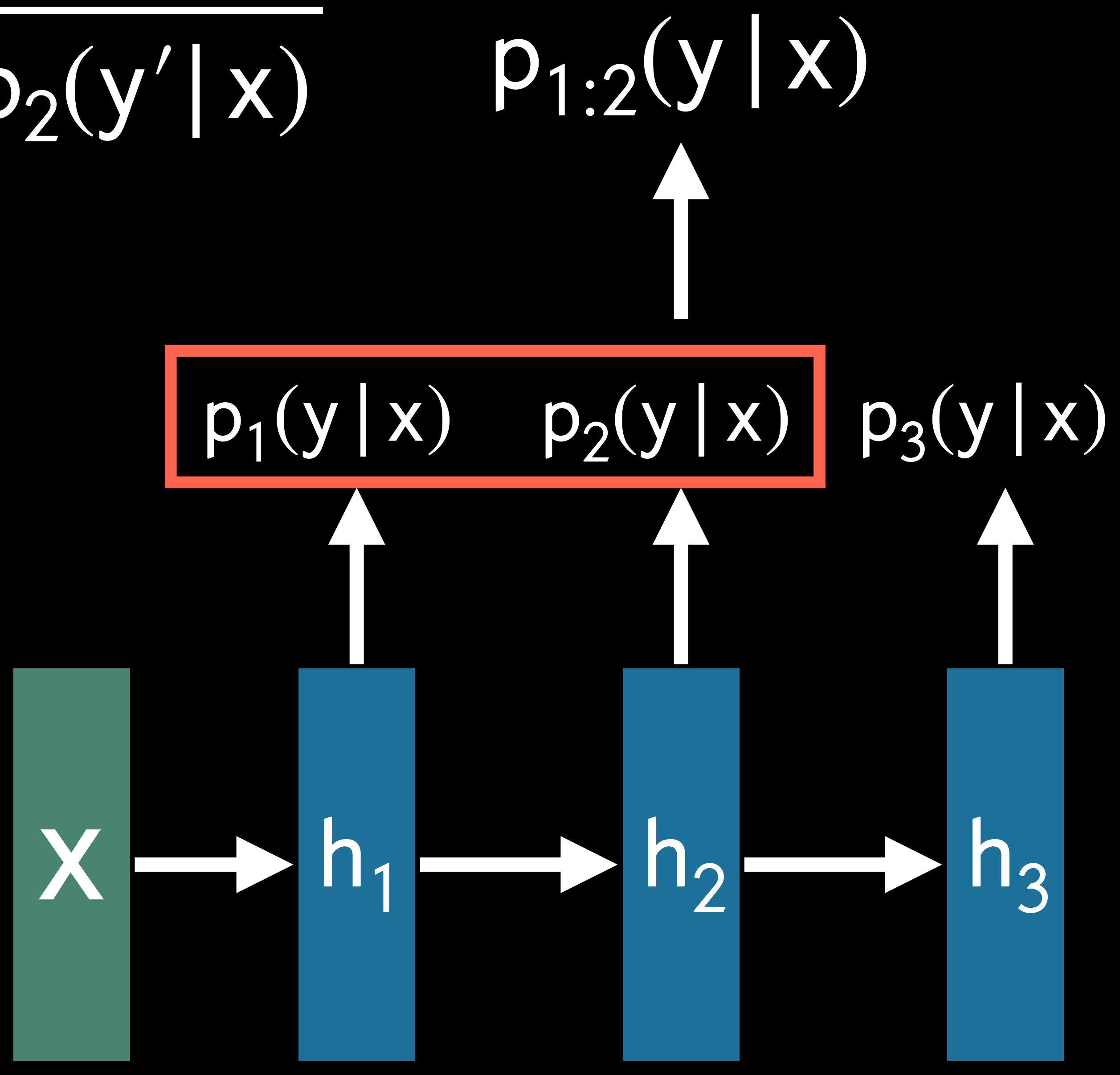


Idea: combine the early-exits via a product of experts

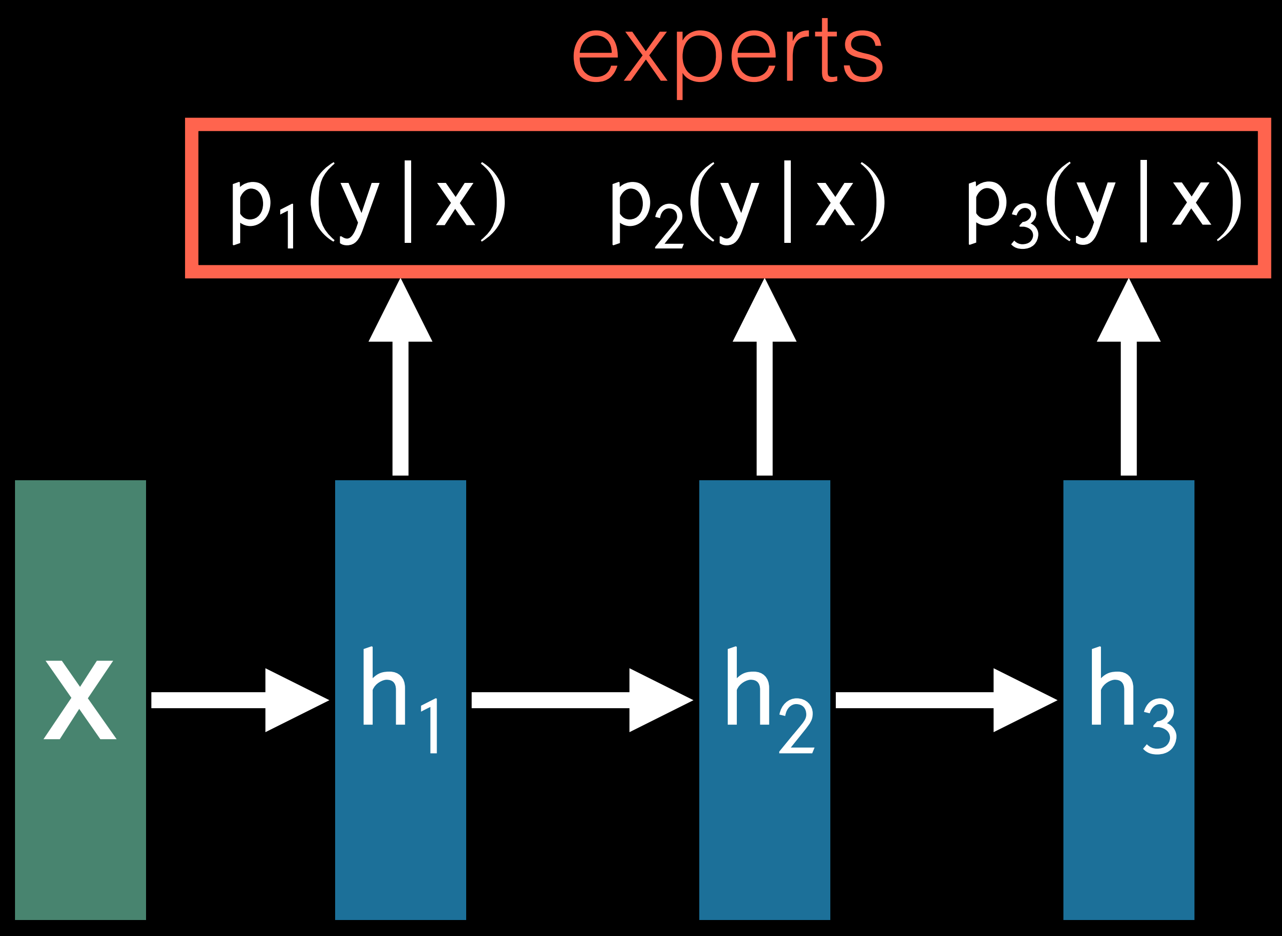


Idea: combine the early-exits via a product of experts

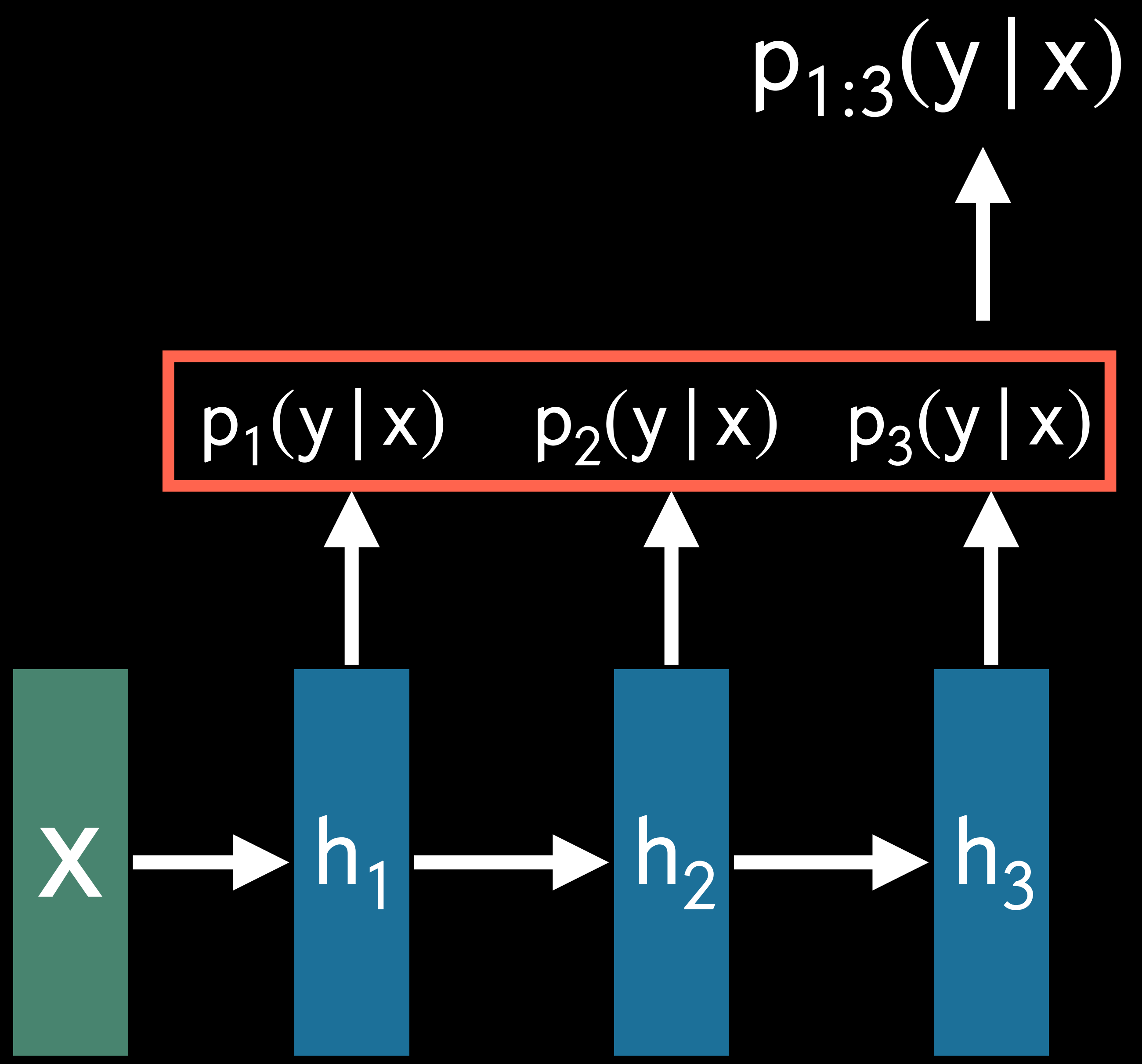
$$p_{1:2}(y|x) = \frac{p_1(y|x) \cdot p_2(y|x)}{\sum_{y'} p_1(y'|x) \cdot p_2(y'|x)}$$



Idea: combine the early-exits via a product of experts

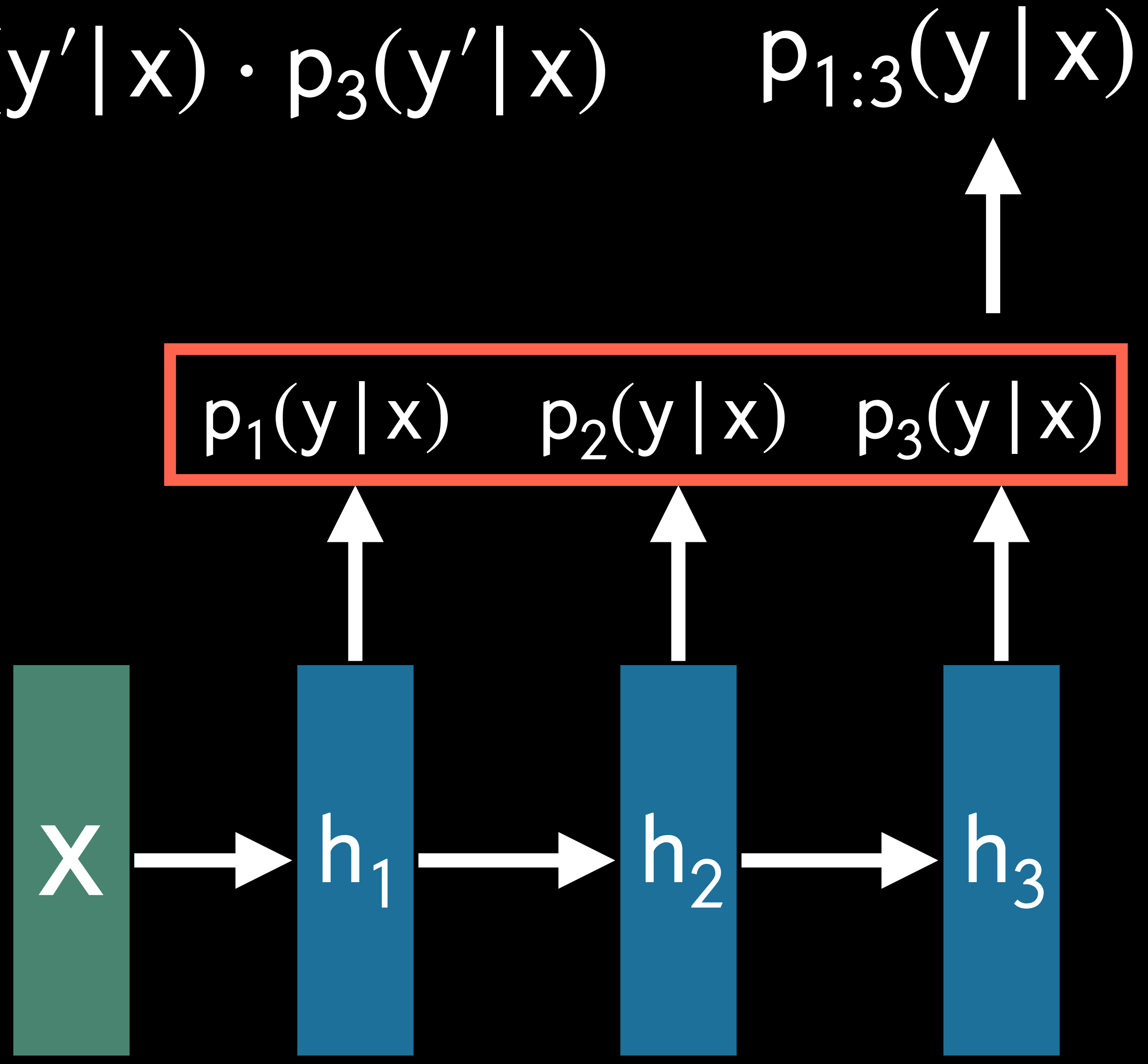


Idea: combine the early-exits via a product of experts



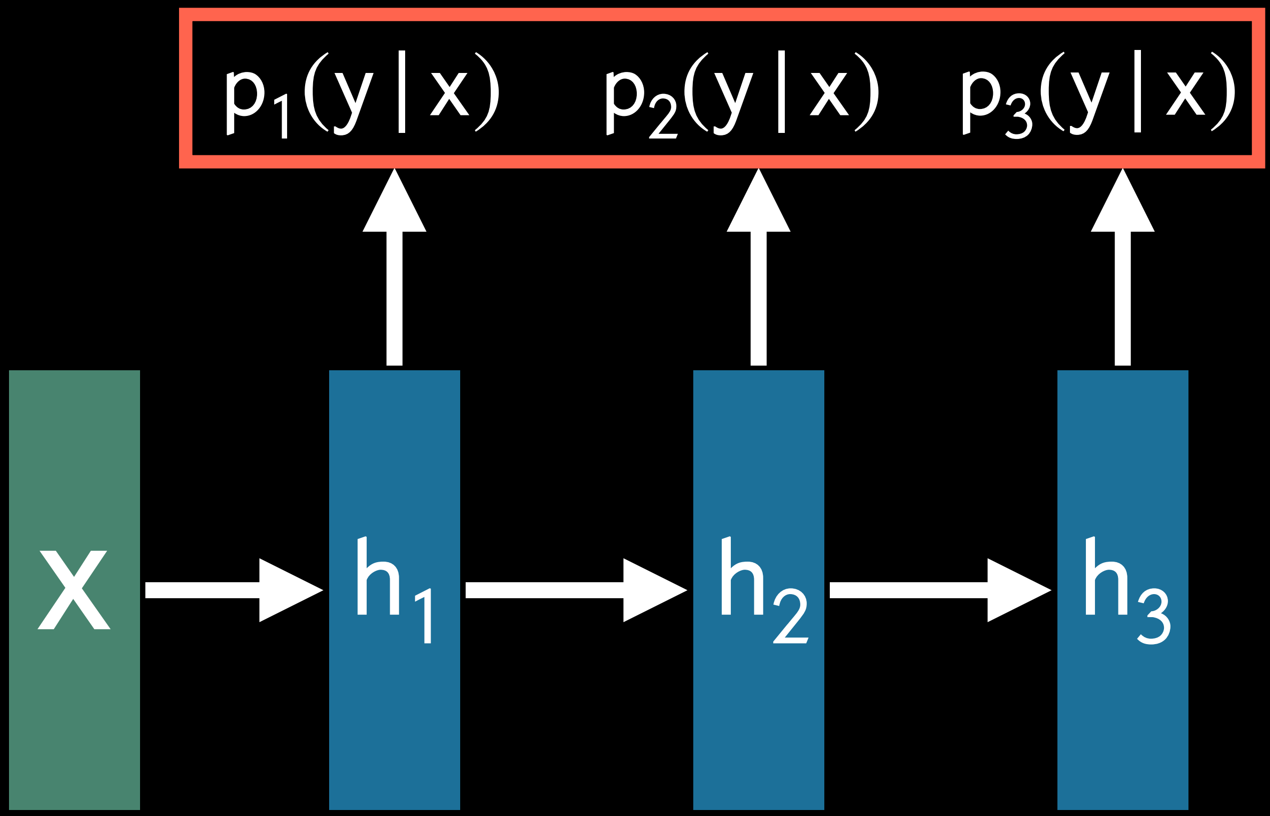
Idea: combine the early-exits via a product of experts

$$p_{1:3}(y|x) = \frac{p_1(y|x) \cdot p_2(y|x) \cdot p_3(y|x)}{\sum_{y'} p_1(y'|x) \cdot p_2(y'|x) \cdot p_3(y'|x)}$$

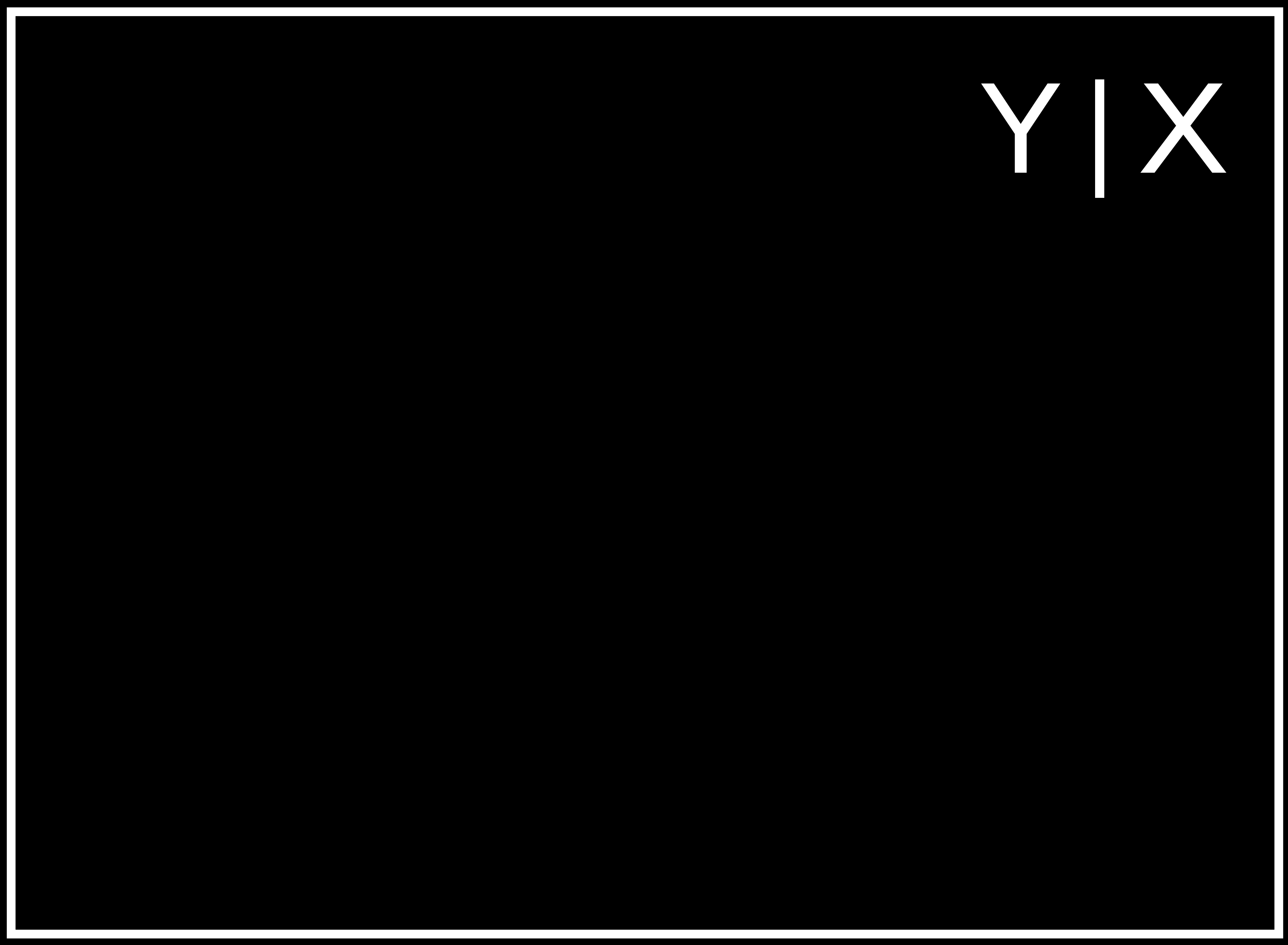


Idea: combine the early-exits via a product of experts

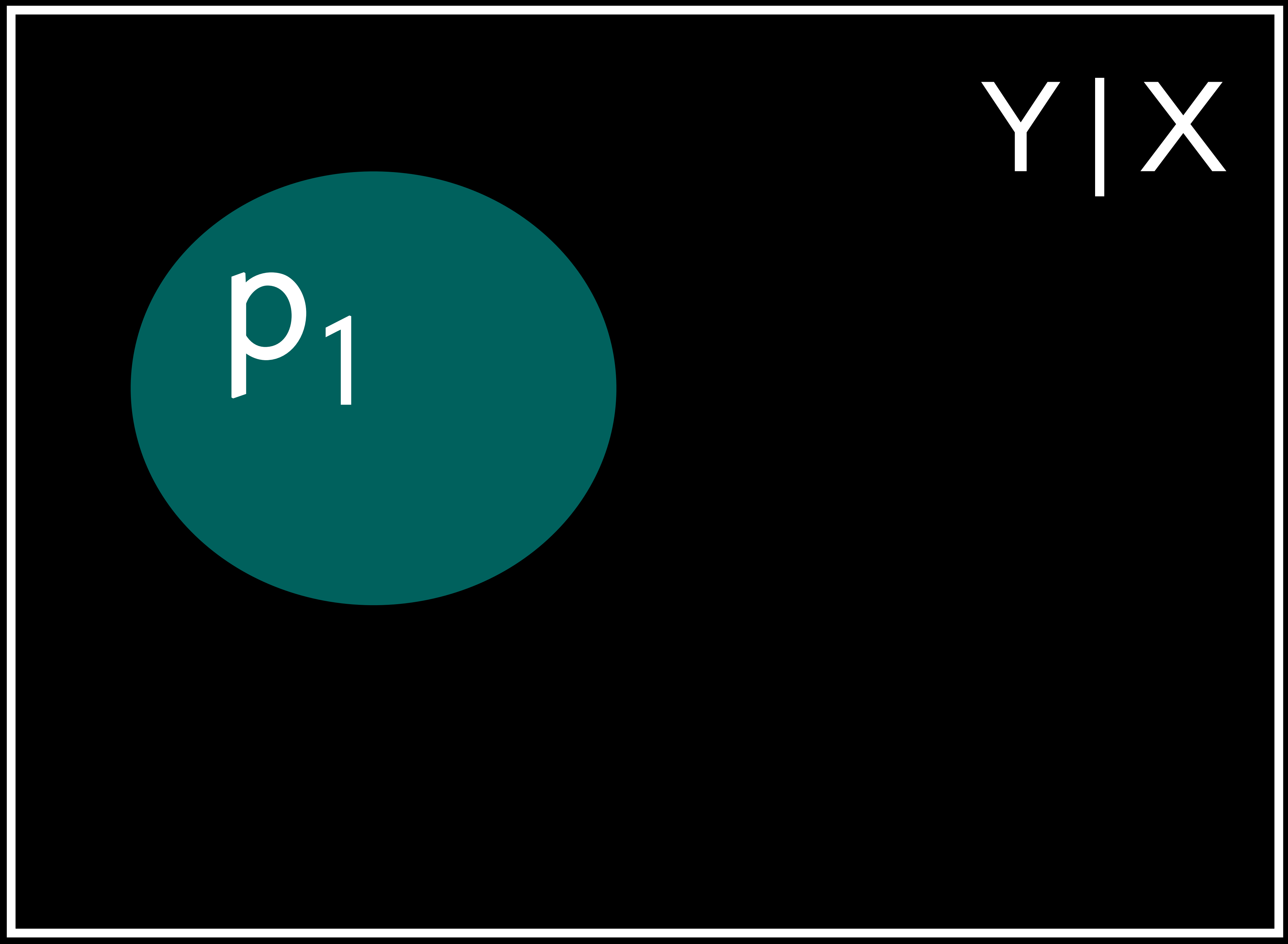
$$p_{1:e}(y \mid x) = \frac{\prod_{j=1}^{e} p_j(y \mid x)}{\sum_{y'} \prod_{j=1}^{e} p_j(y' \mid x)}$$



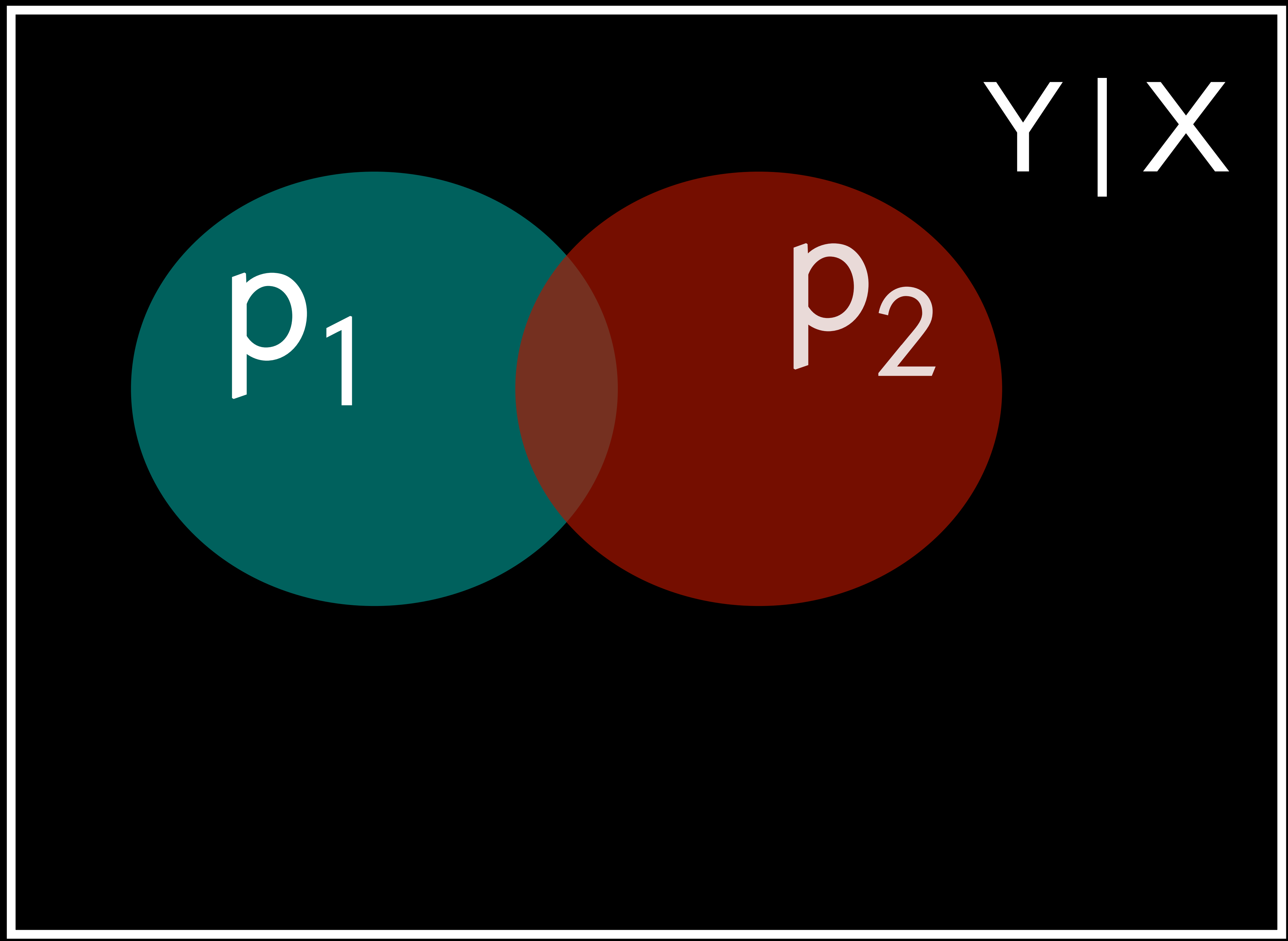
Idea: combine the early-exits via a product of experts



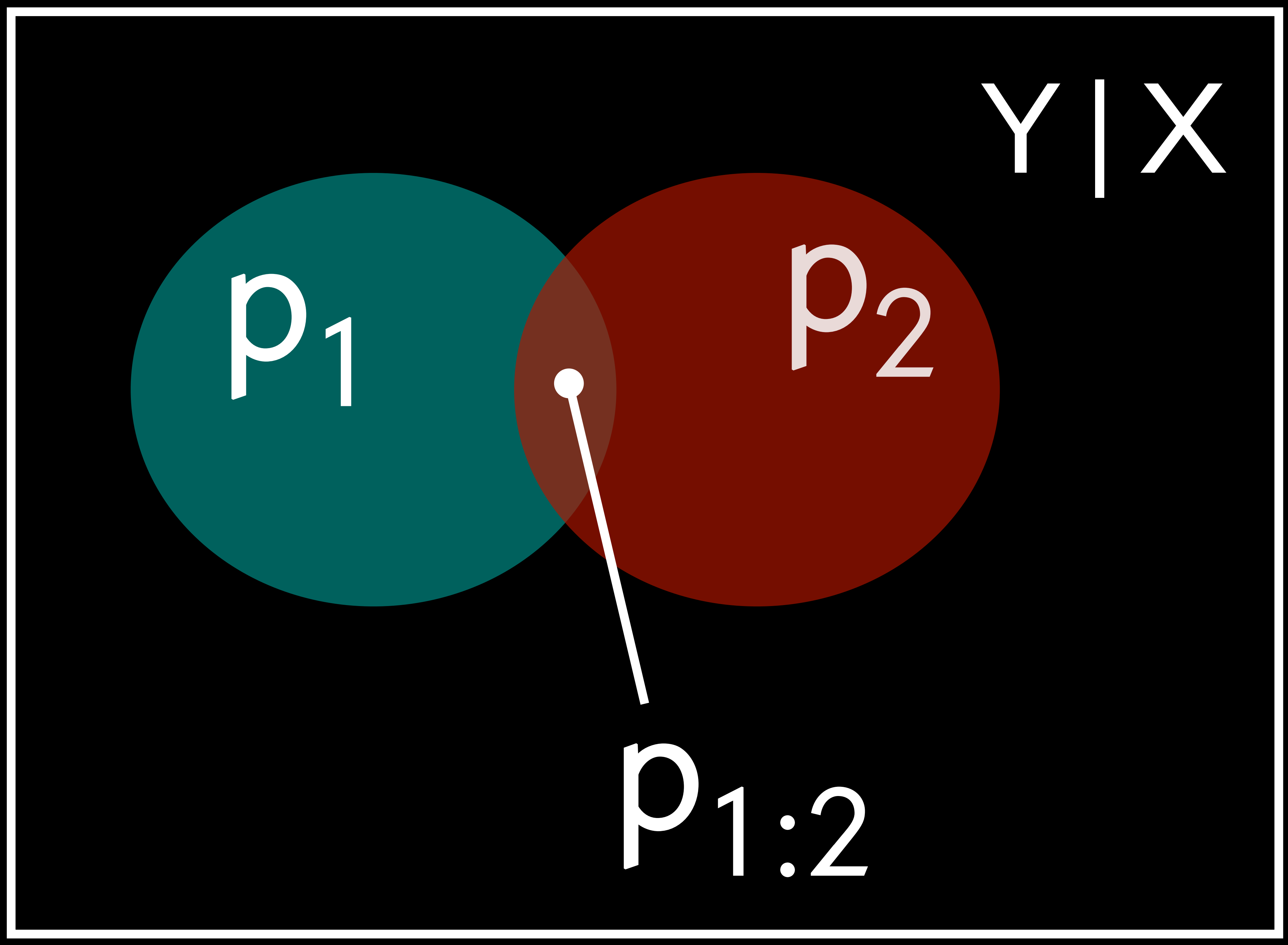
Idea: combine the early-exits via a product of experts



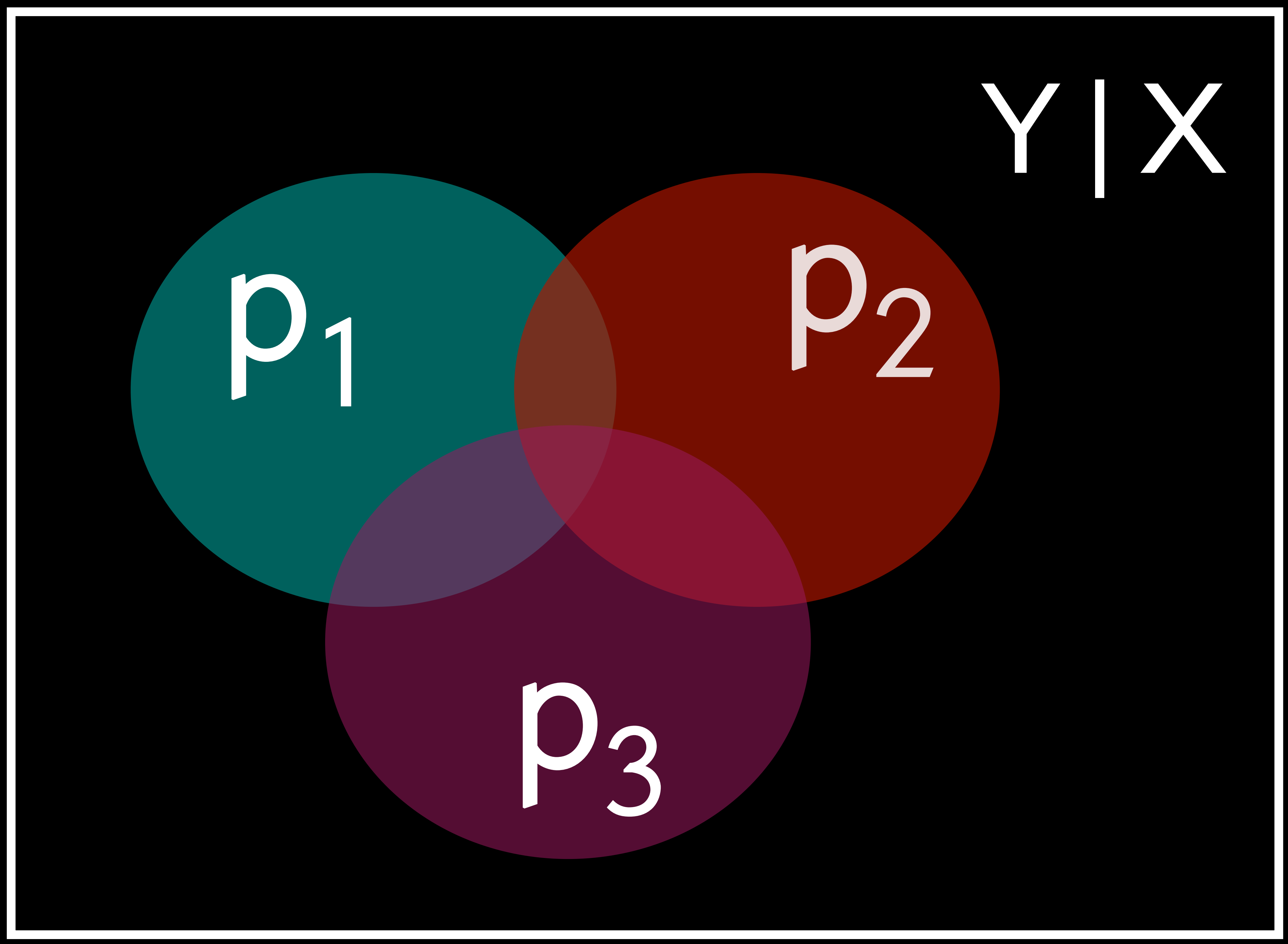
Idea: combine the early-exits via a product of experts



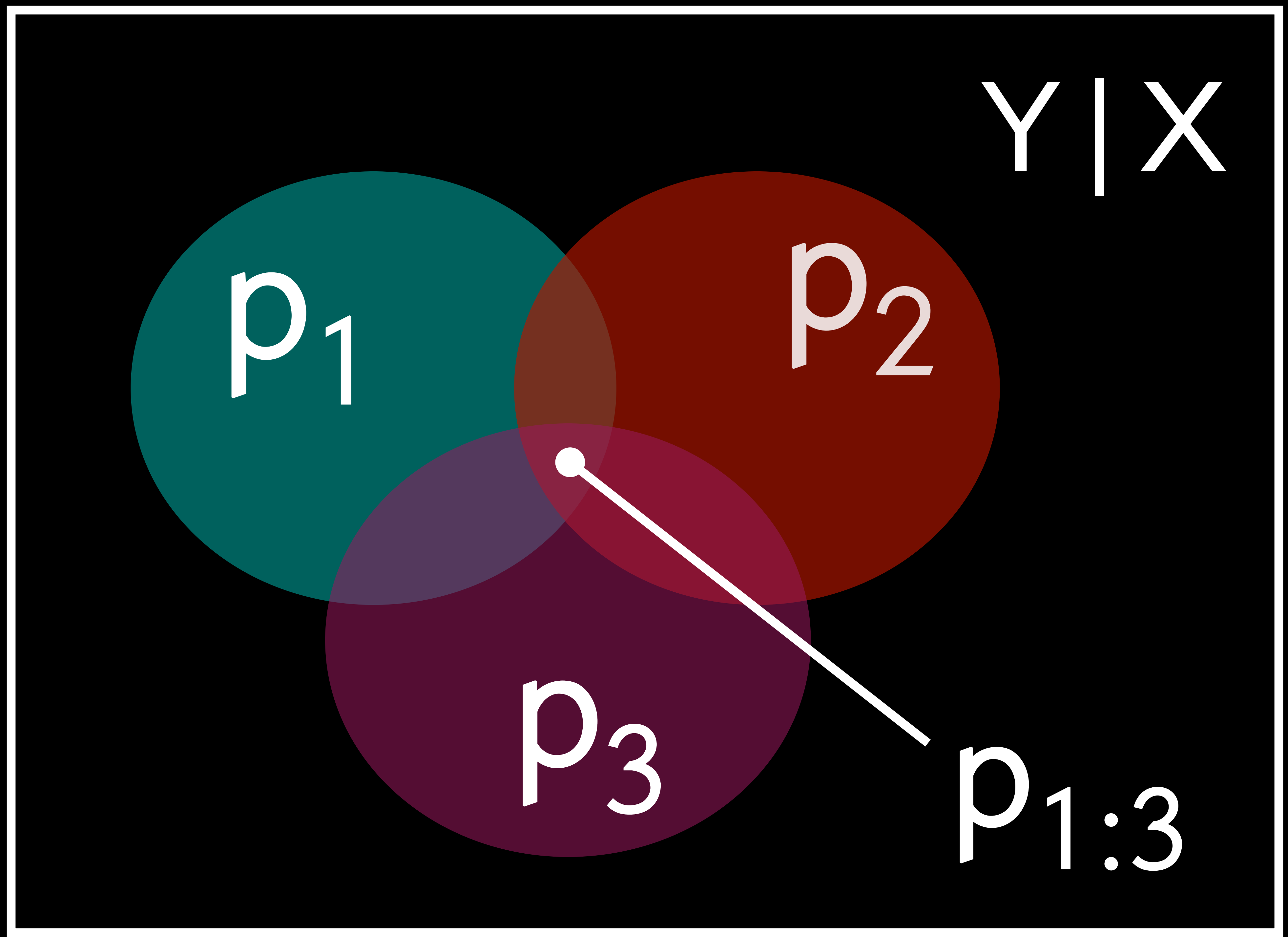
Idea: combine the early-exits via a product of experts



Idea: combine the early-exits via a product of experts



Idea: combine the early-exits via a product of experts



Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Ideal case: binary one-vs-rest.

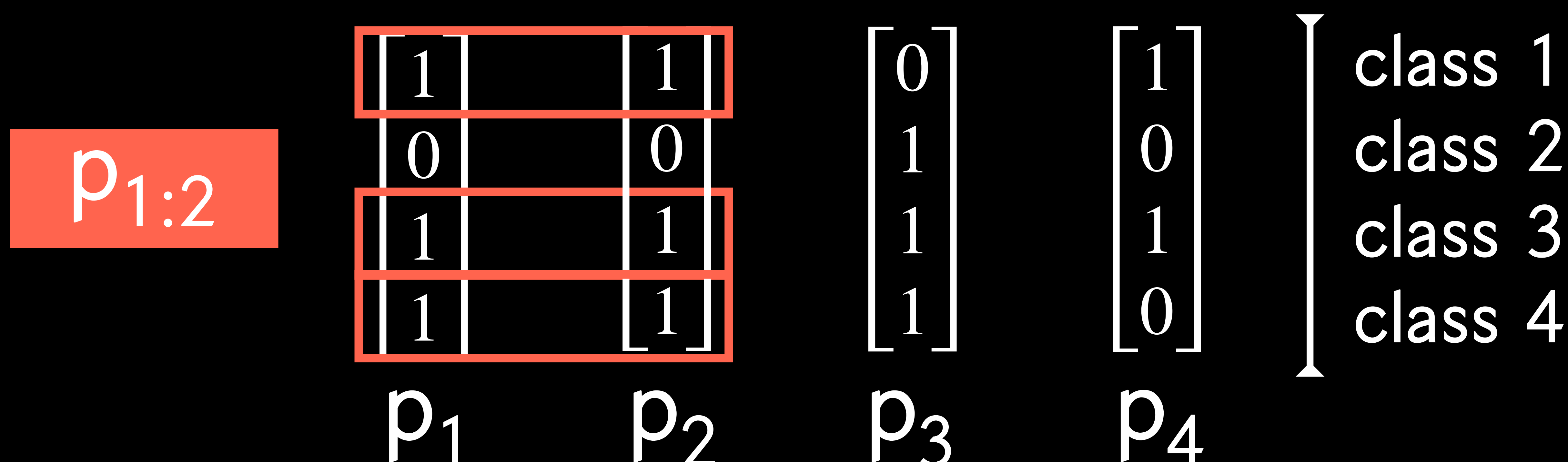
$$\underset{p_1}{\begin{bmatrix}1\\0\\1\\1\end{bmatrix}} \quad \underset{p_2}{\begin{bmatrix}1\\0\\1\\1\end{bmatrix}} \quad \underset{p_3}{\begin{bmatrix}0\\1\\1\\1\end{bmatrix}} \quad \underset{p_4}{\begin{bmatrix}1\\0\\1\\0\end{bmatrix}}$$

class 1
class 2
class 3
class 4

Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Ideal case: binary one-vs-rest.



Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Ideal case: binary one-vs-rest.

$p_{1:3}$

$$p_1 = \begin{bmatrix}1\\0\\1\\1\end{bmatrix} \quad p_2 = \begin{bmatrix}1\\0\\1\\1\end{bmatrix} \quad p_3 = \begin{bmatrix}0\\1\\1\\1\end{bmatrix} \quad p_4 = \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

class 1
class 2
class 3
class 4

Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Ideal case: binary one-vs-rest.

$p_{1:4}$

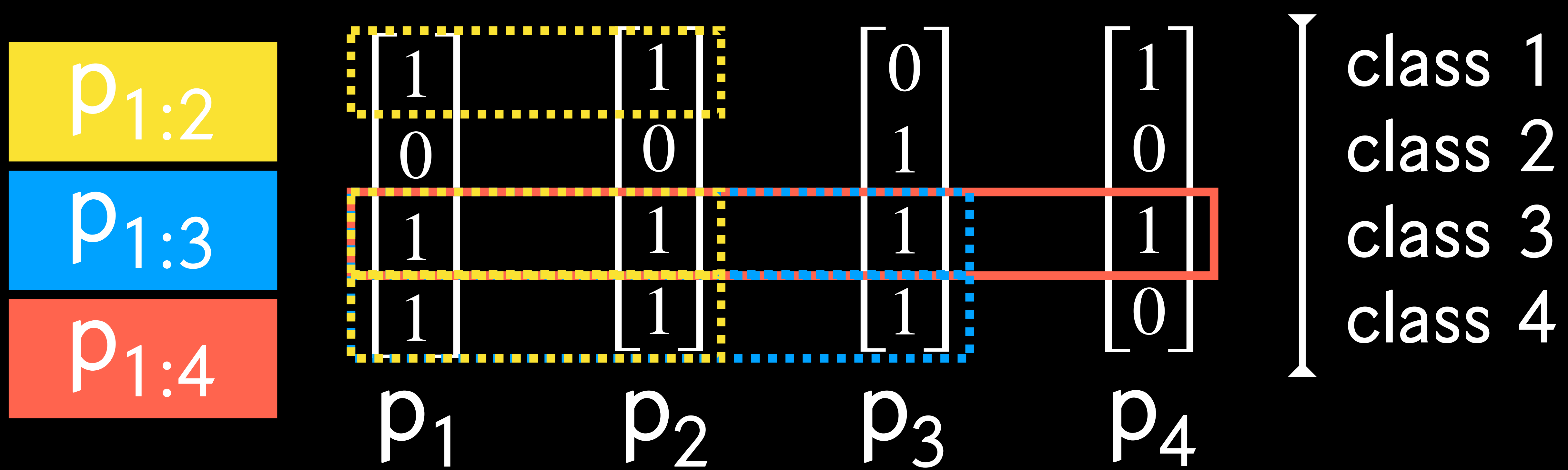
$$\underset{p_1}{\begin{bmatrix}1\\0\\1\\1\end{bmatrix}} \quad \underset{p_2}{\begin{bmatrix}1\\0\\1\\1\end{bmatrix}} \quad \underset{p_3}{\begin{bmatrix}0\\1\\1\\1\end{bmatrix}} \quad \underset{p_4}{\begin{bmatrix}1\\0\\1\\0\end{bmatrix}}$$

class 1
class 2
class 3
class 4

Idea: combine the early-exits via a product of experts

One catch: exit distributions must have finite (or quickly decaying) support to bound influence of (e+1)th expert.

Ideal case: binary one-vs-rest.



Implementation with ReLUs

$$p_{1:e}(y|x) = \frac{\prod_{j=1}^{e} \max\left(0,\ f_{j,y}(x)\right)}{\sum_{y'} \prod_{j=1}^{e} \max\left(0,\ f_{j,y'}(x)\right)}$$

$f_{j,y}(x)$ is logit for yth class at jth exit

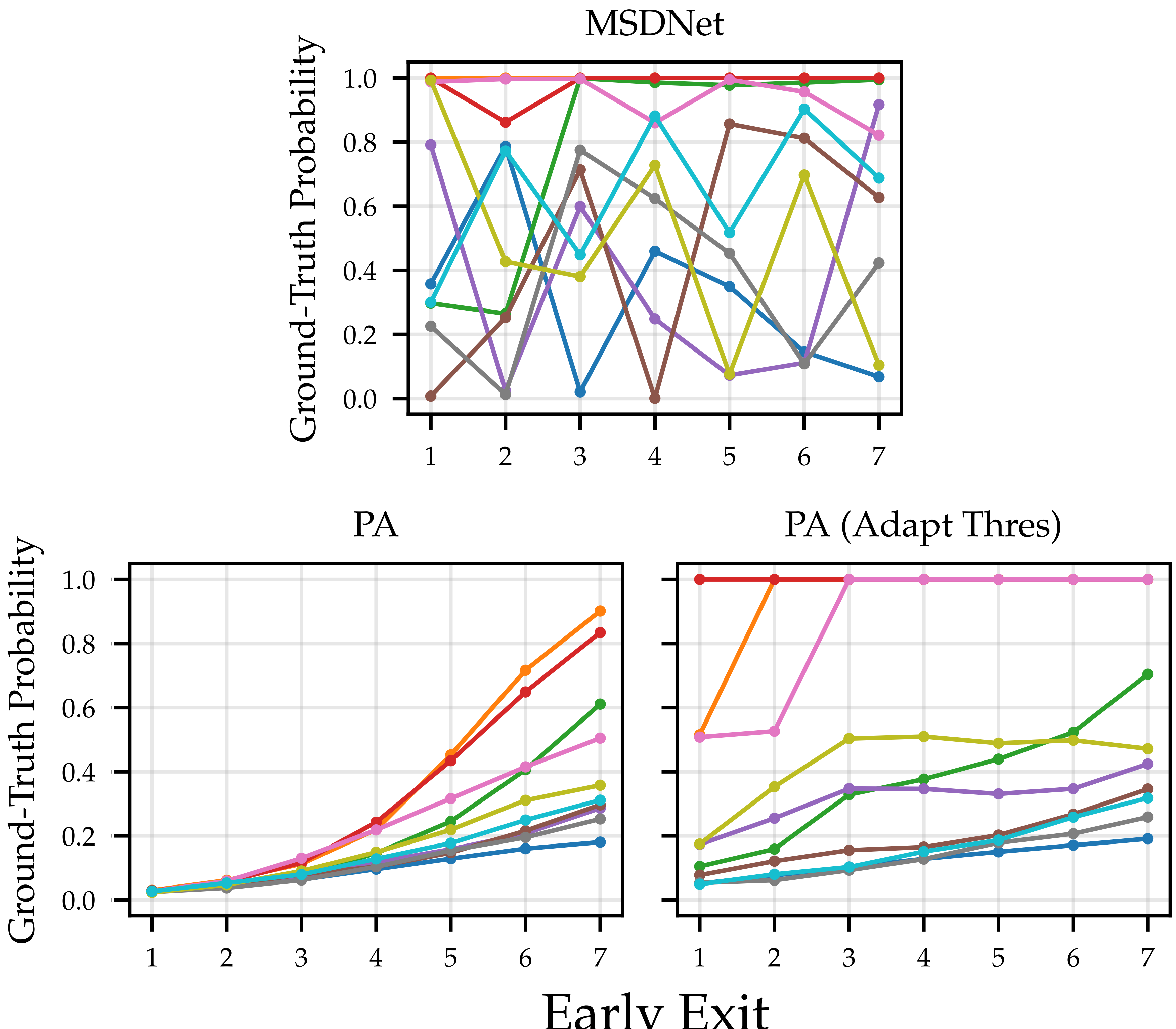
Implementation with ReLUs

$$p_{1:e}(y|x) = \frac{\prod_{j=1}^{e} \max\left(0,\ f_{j,y}(x)\right)}{\sum_{y'} \prod_{j=1}^{e} \max\left(0,\ f_{j,y'}(x)\right)}$$

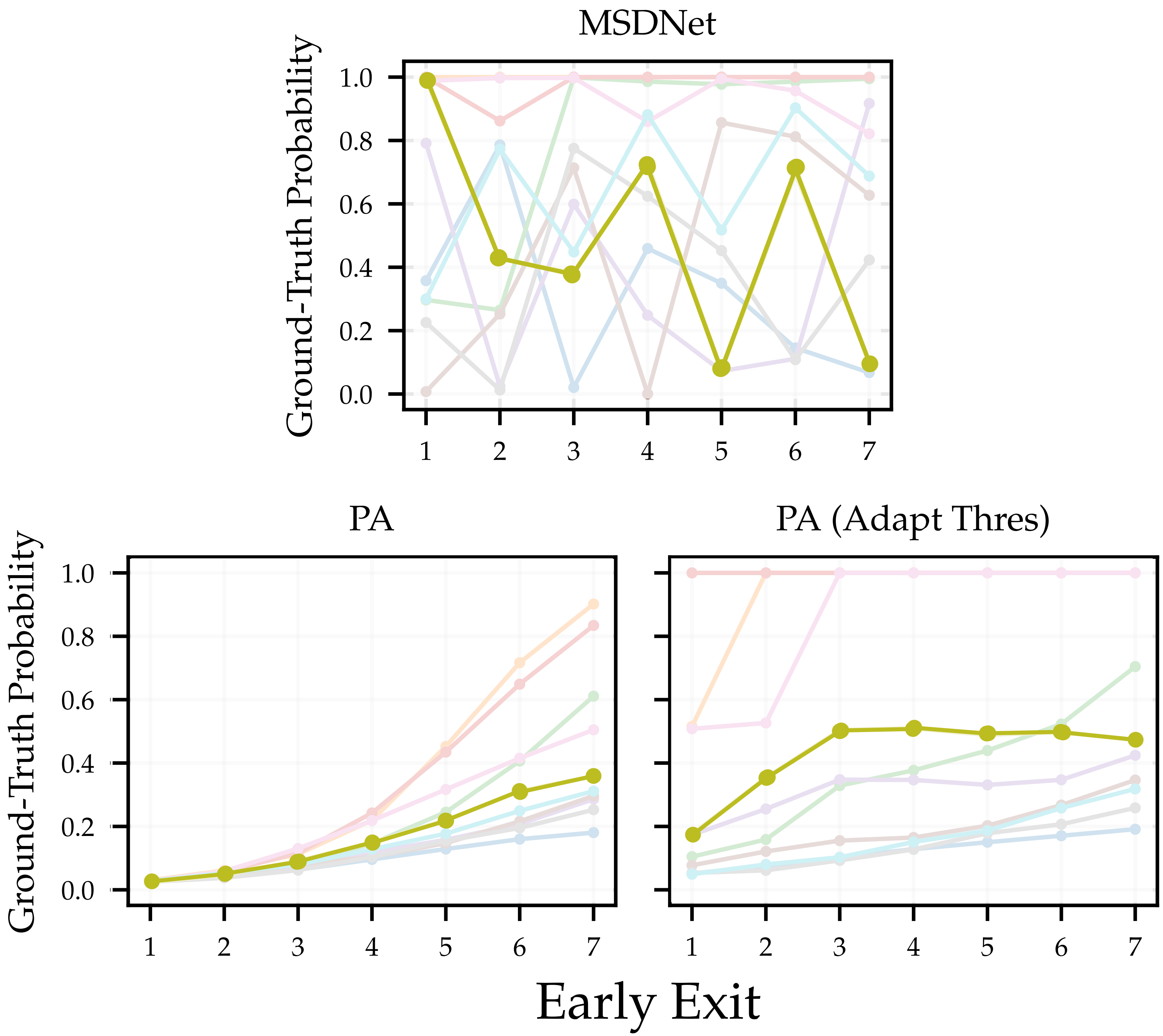
Clipping logits controls deviation from perfect monotonicity.

We apply this transformation post-hoc!

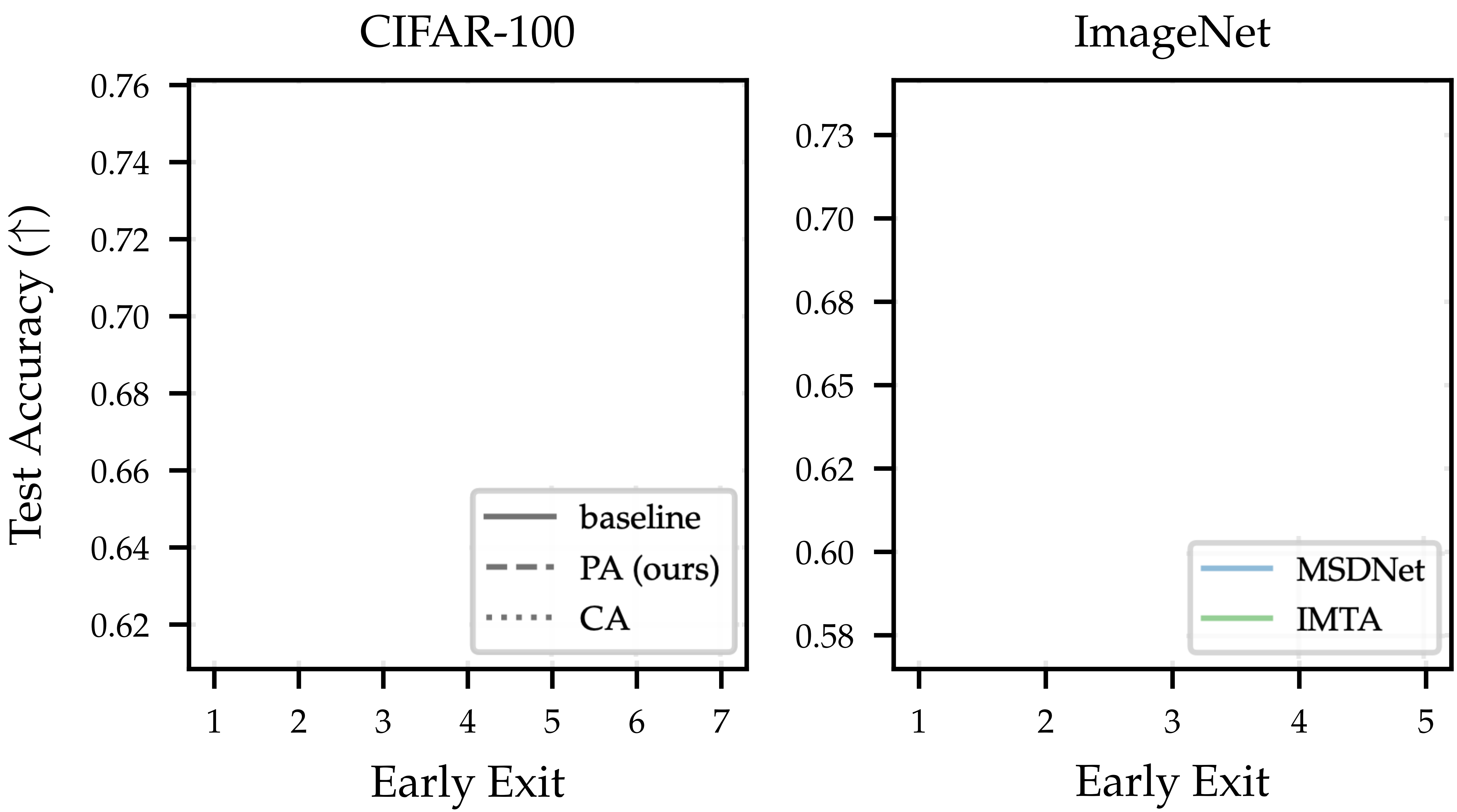
Monotonicity: CIFAR-100



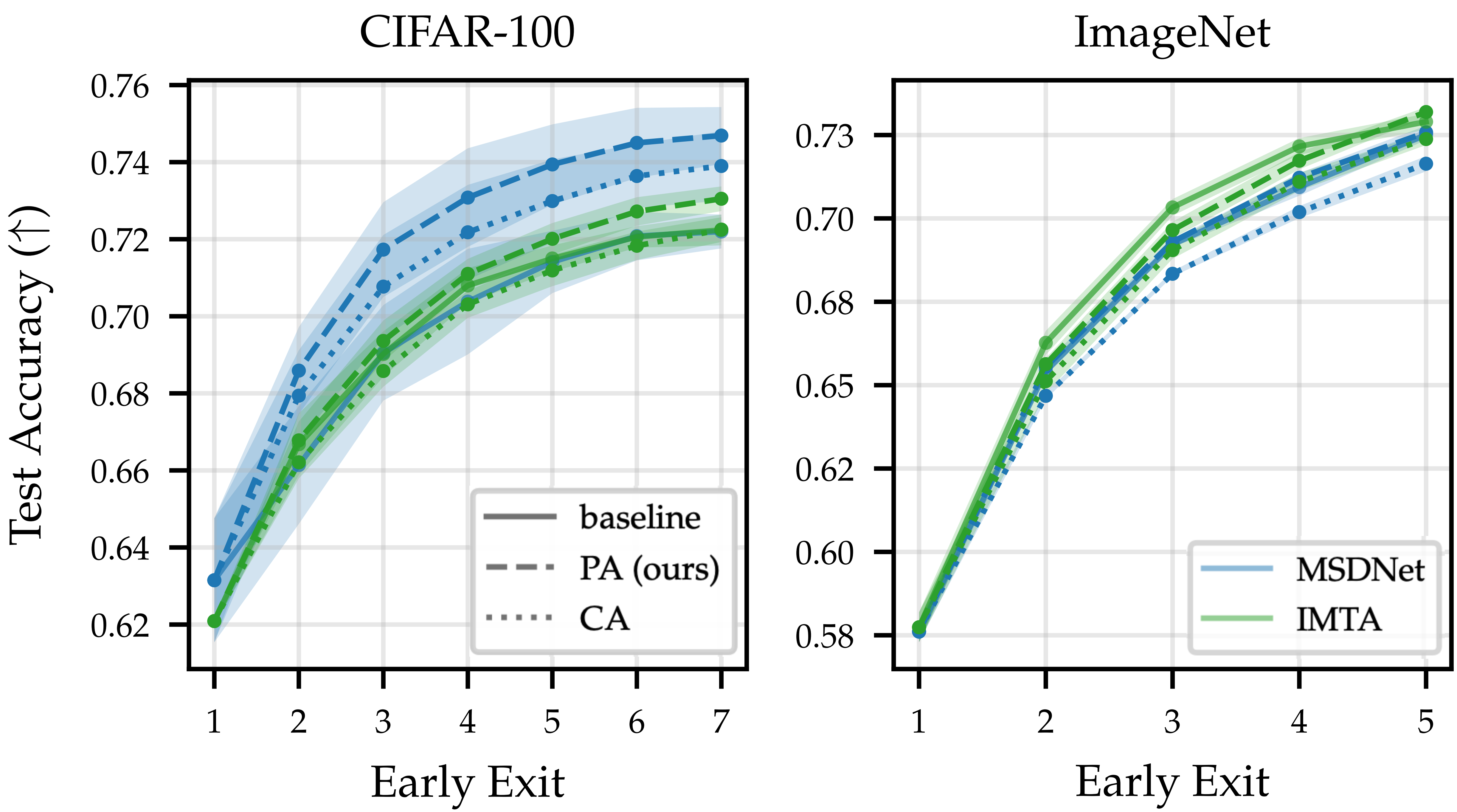
Monotonicity: CIFAR-100



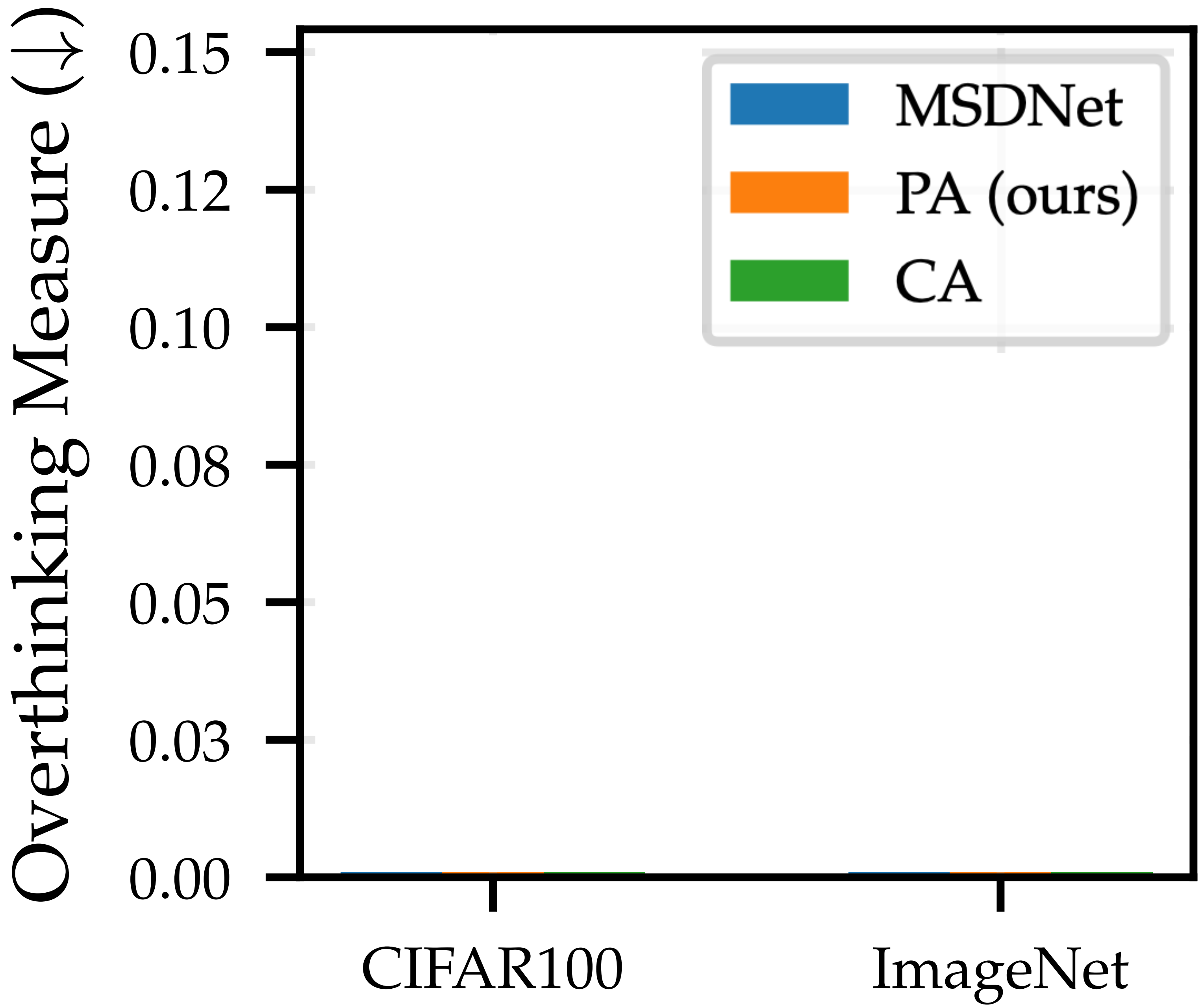
Accuracy: CIFAR-100 & ImageNet



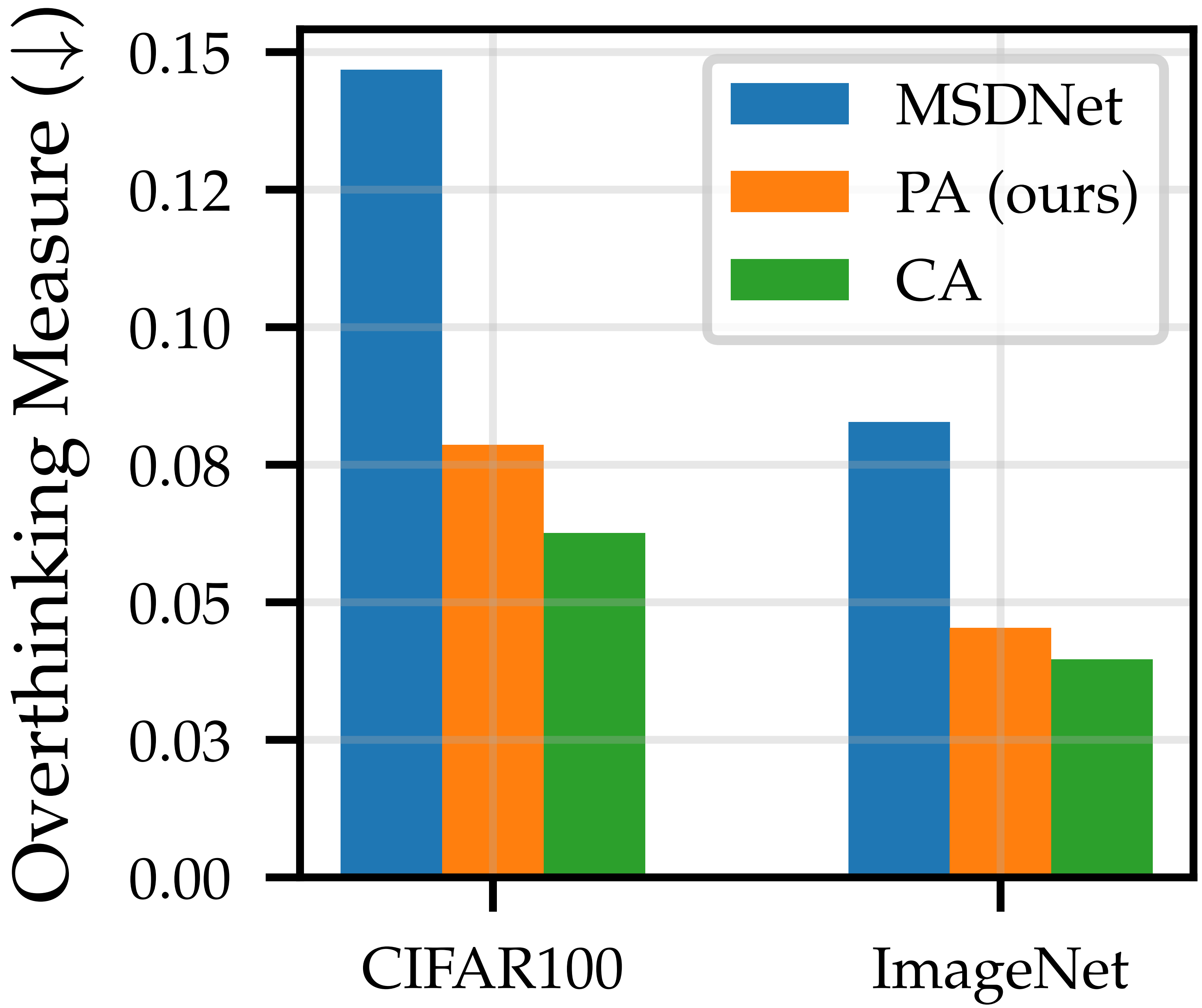
Accuracy: CIFAR-100 & ImageNet



Overthinking: CIFAR-100 & ImageNet

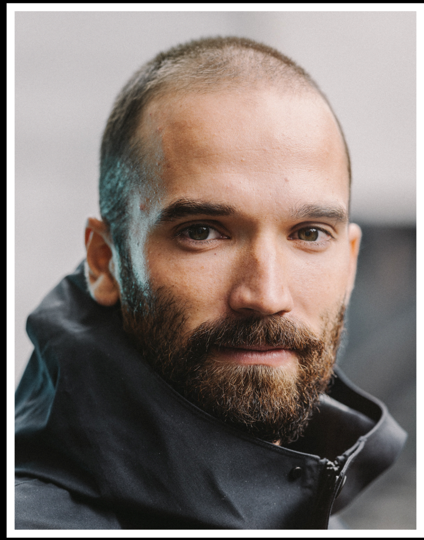


Overthinking: CIFAR-100 & ImageNet



*Doesn't mean that overall accuracy is improved by this amount since our model makes more mistakes at intermediate exits.

Ensuring consistency across exits in predictive uncertainty estimates



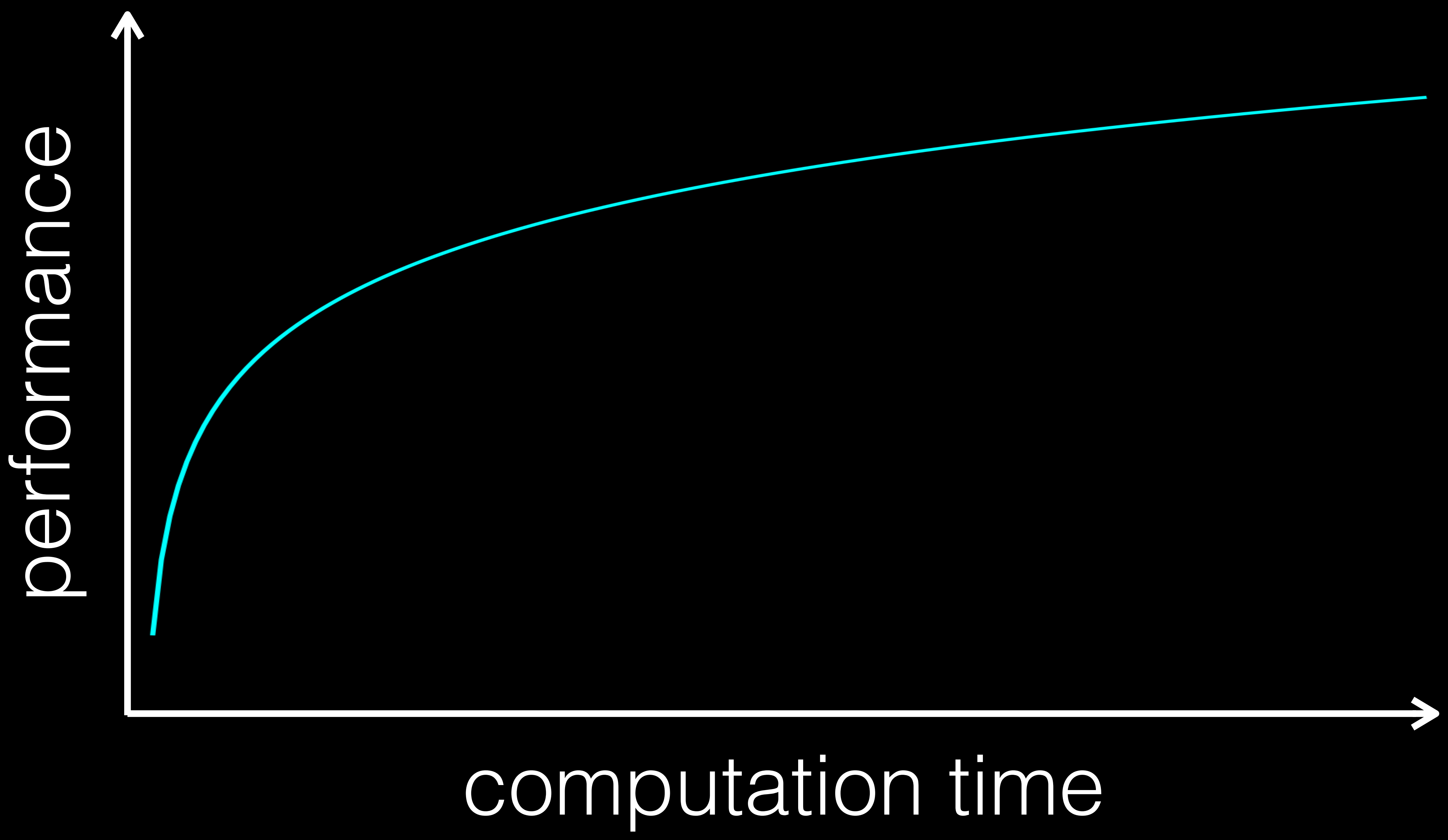
Metod
Jazbec

Dan
Zhang

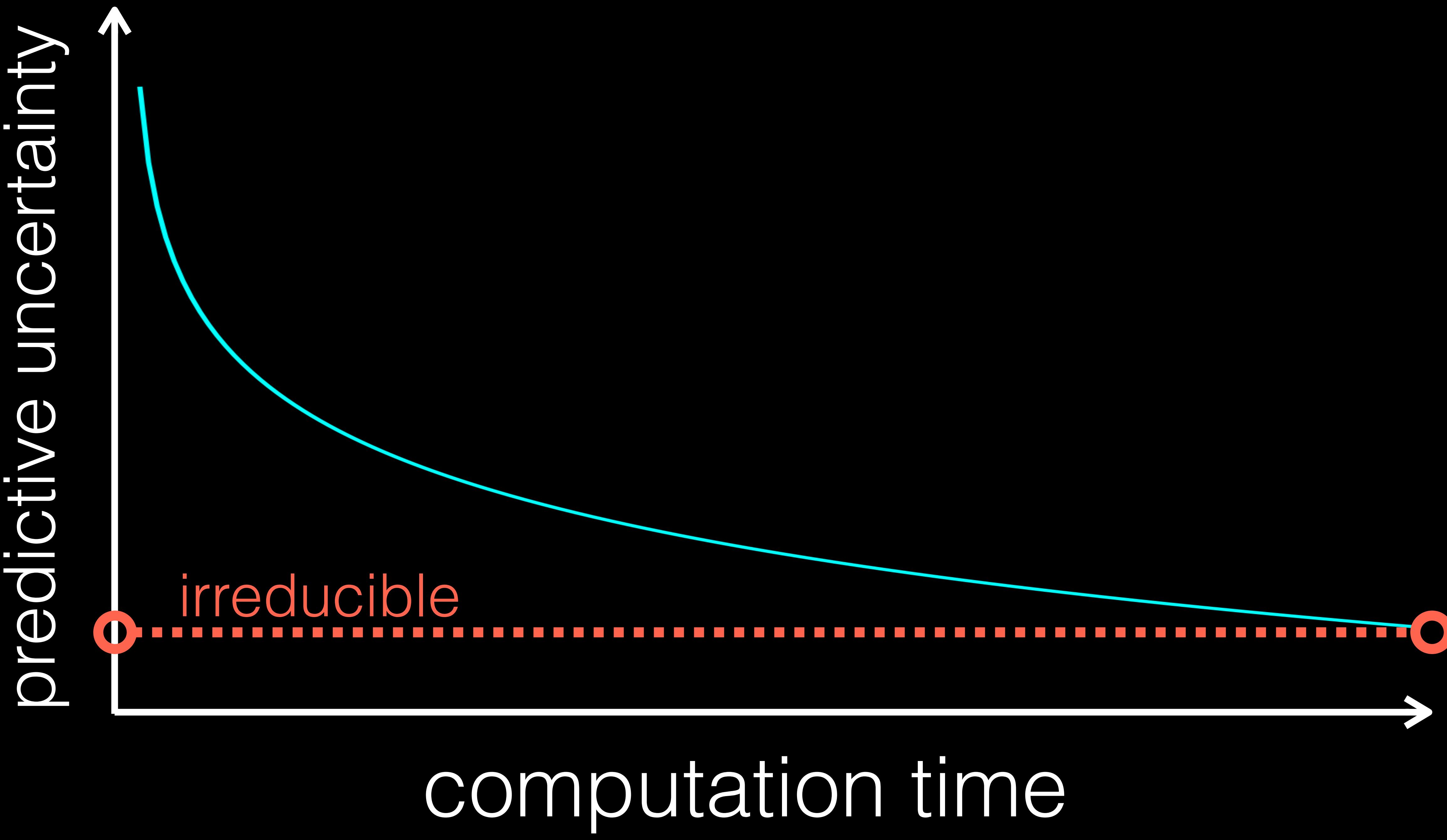
Patrick
Forré

Stephan
Mandt

Anytime Models



Anytime Uncertainty

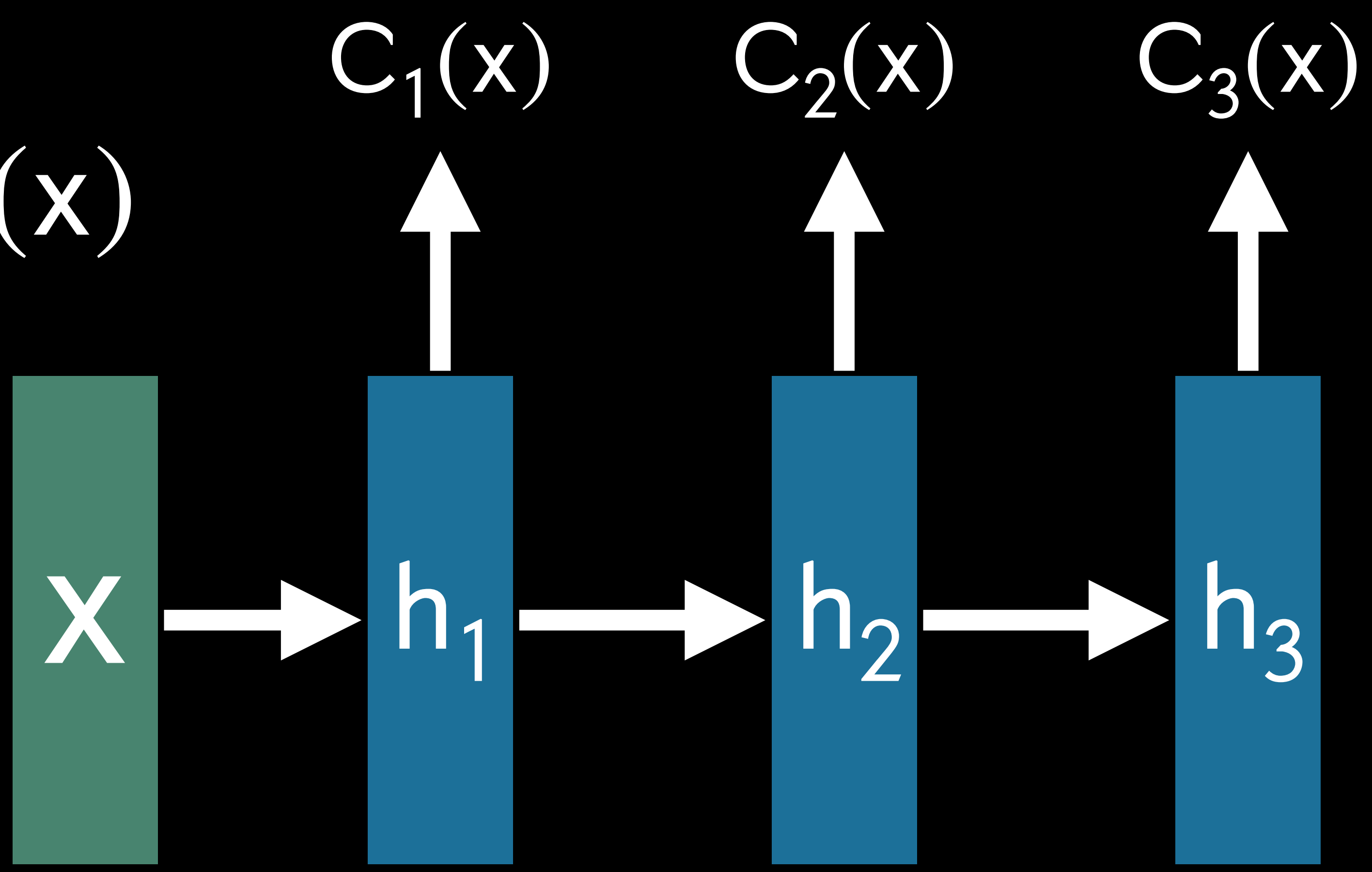


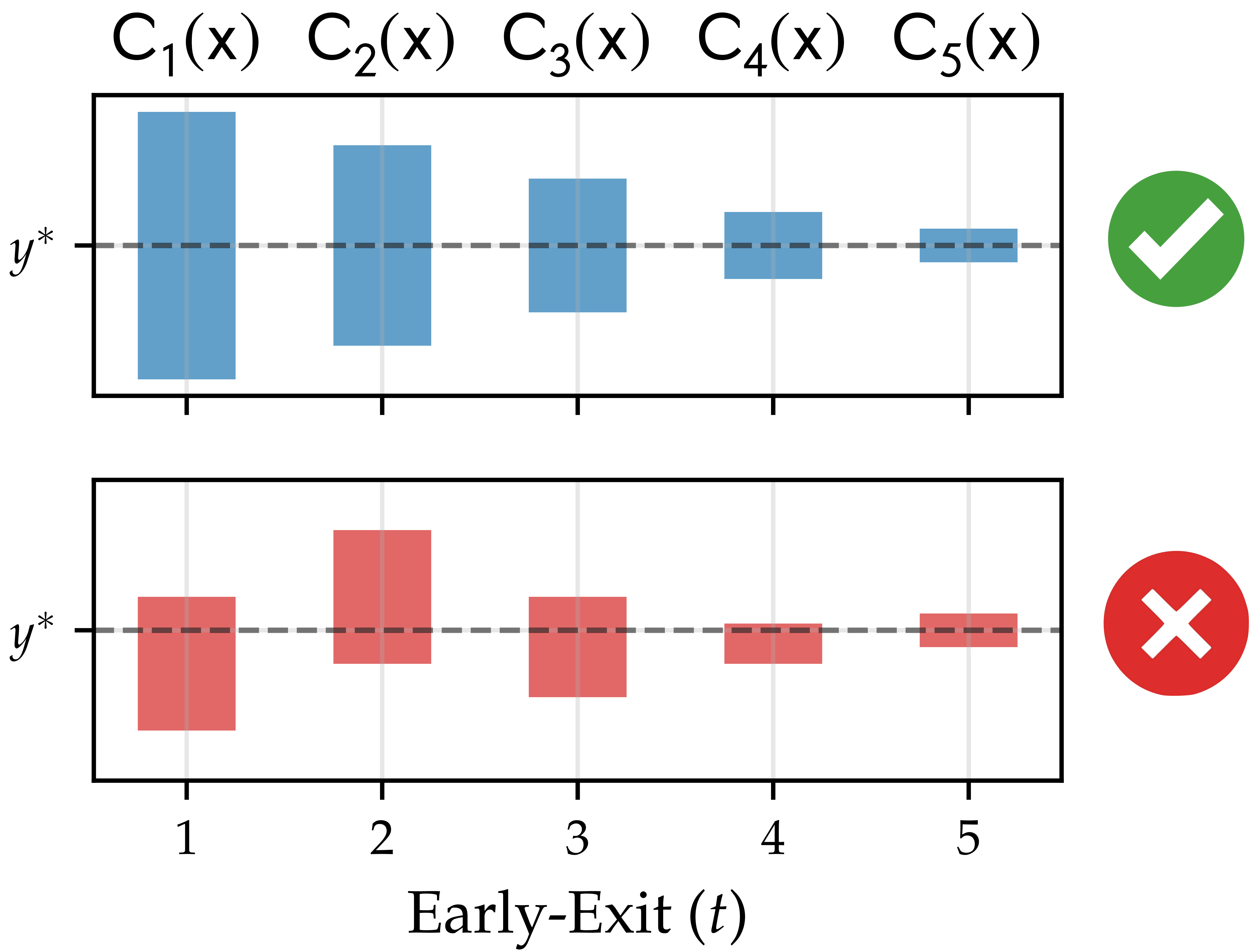
Anytime Uncertainty Estimation

We want nested, non-increasing prediction intervals across exits.

consistency:

$$C_1(x) \subseteq C_2(x) \subseteq C_3(x)$$



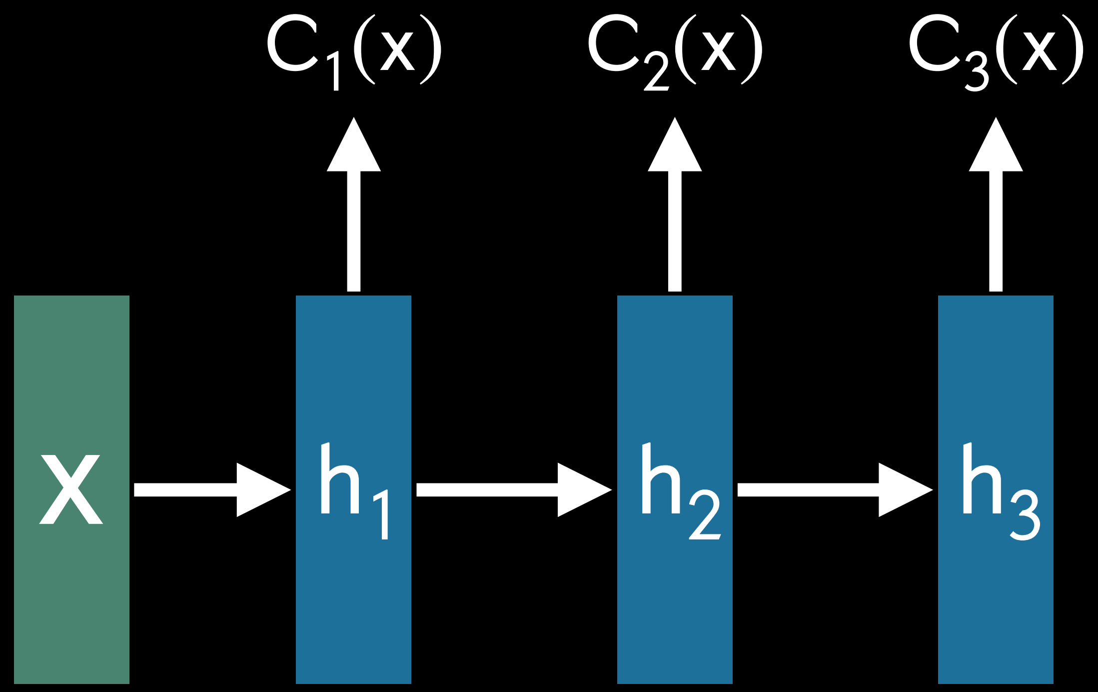

C1(x)
C2(x)
C3(x)
C4(x)
C5(x)
y*
y*
1
2
3
4
5
Early-Exit (t)

Anytime-Valid Confidence Sequences

We construct an *anytime-valid confidence sequence* across the exits.

$$\mathbb{P}\left(\forall t,\ y^* \in C_t(x)\right) \geq 1 - \alpha$$

Due to approximations, we can only hope to achieve this for large datasets (and if y is from the training distribution).



[Robbins, AMS 1970]

Anytime-Valid Confidence Sequences

Derived from the following predictive-likelihood martingale:

$$R_t(y) = \prod_{e=1}^{t} \frac{p_e(y \mid x, \mathfrak{D})}{p_e(y \mid x, \hat{\theta}_e)} \quad \hat{\theta}_e \sim p\left(\theta_e \mid x, \mathfrak{D}\right)$$

Anytime-Valid Confidence Sequences

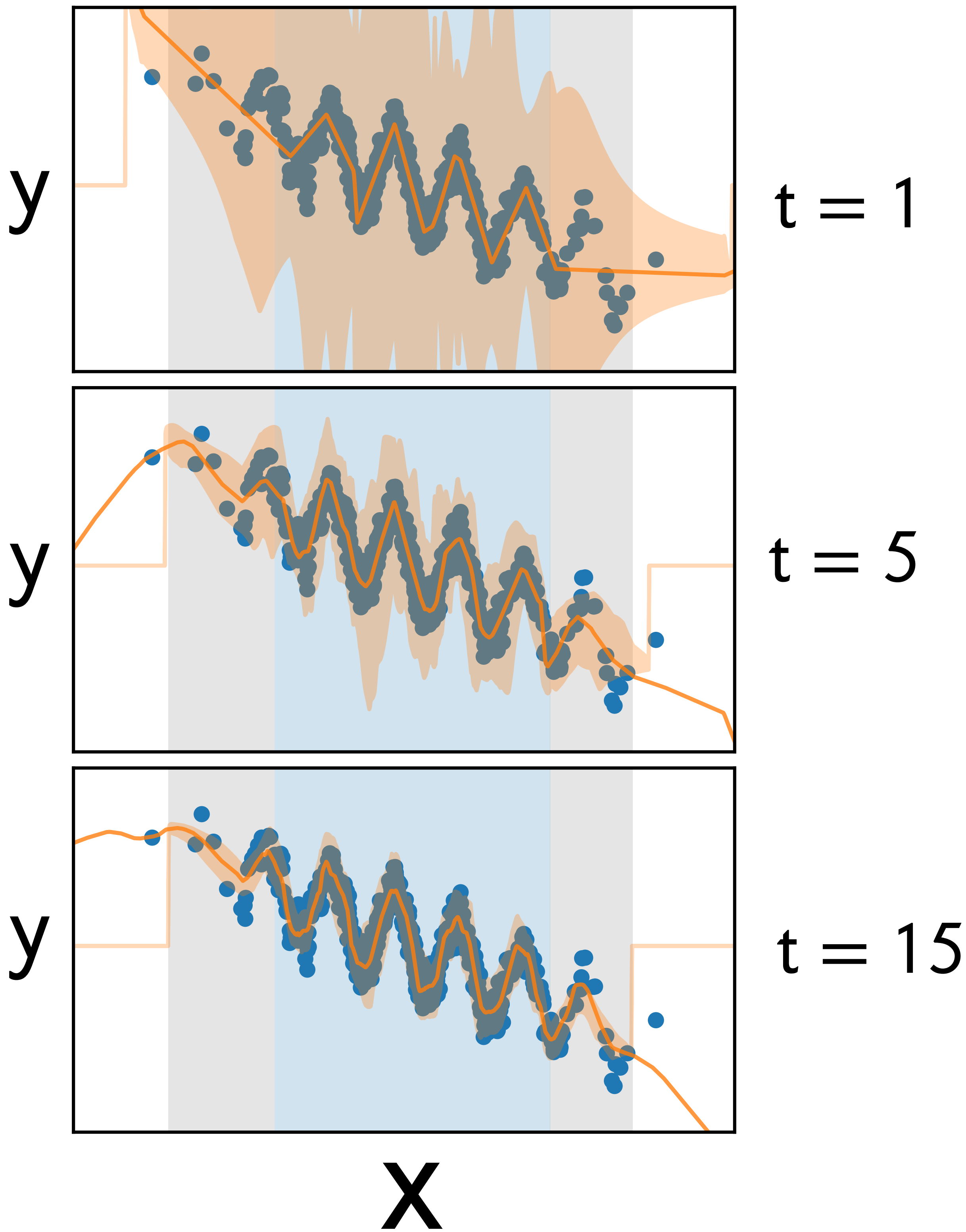
Derived from the following predictive-likelihood martingale:

$$R_t(y) = \prod_{e=1}^{t} \frac{p_e(y \mid x, \mathfrak{D})}{p_e(y \mid x, \hat{\theta}_e)} \quad \hat{\theta}_e \sim p\left(\theta_e \mid x, \mathfrak{D}\right)$$

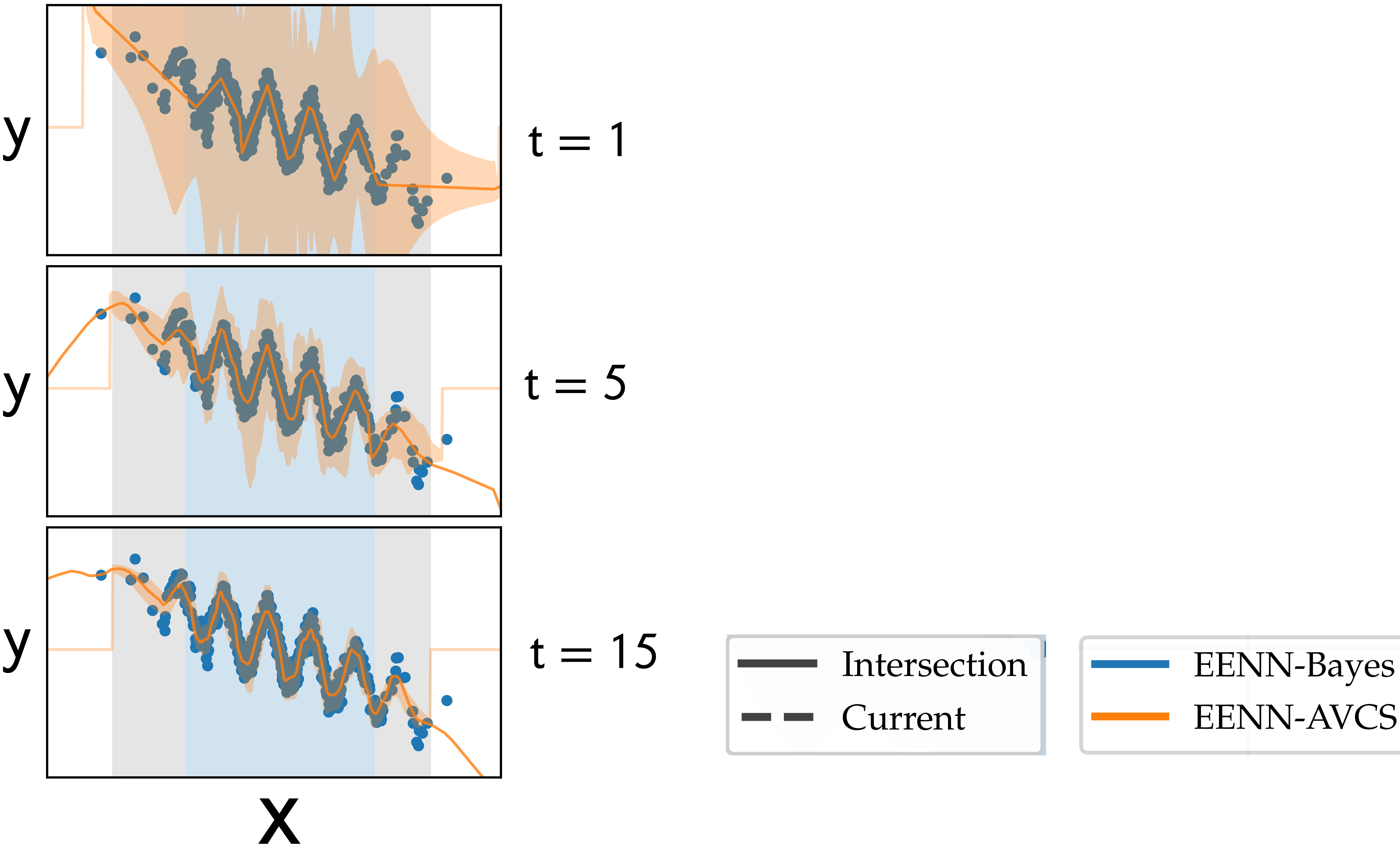
Construct set at time t as:

$$C_t(x) = \left\{ y \in Y \mid R_t(y) \leq 1/\alpha \right\}$$

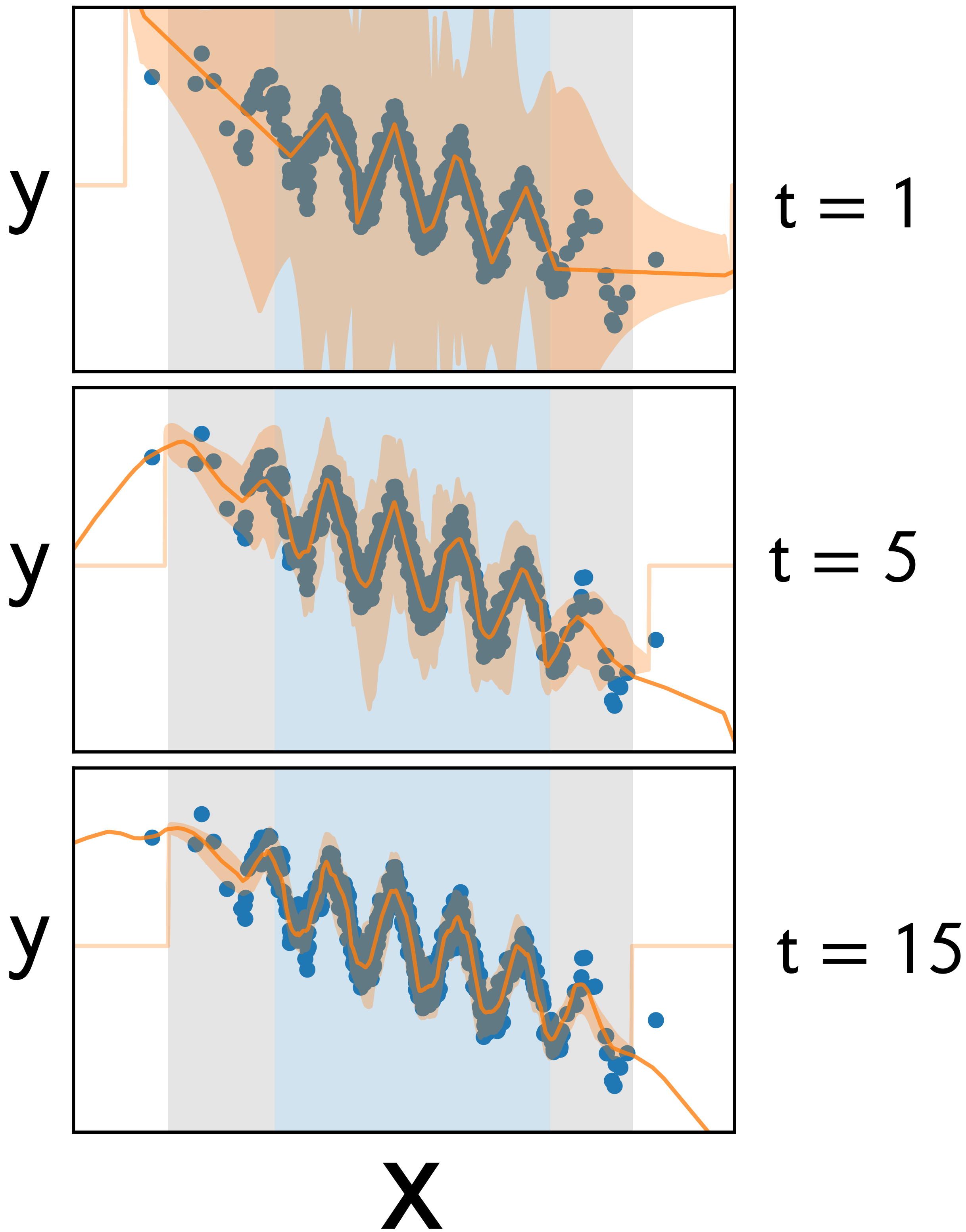
Regression Simulation

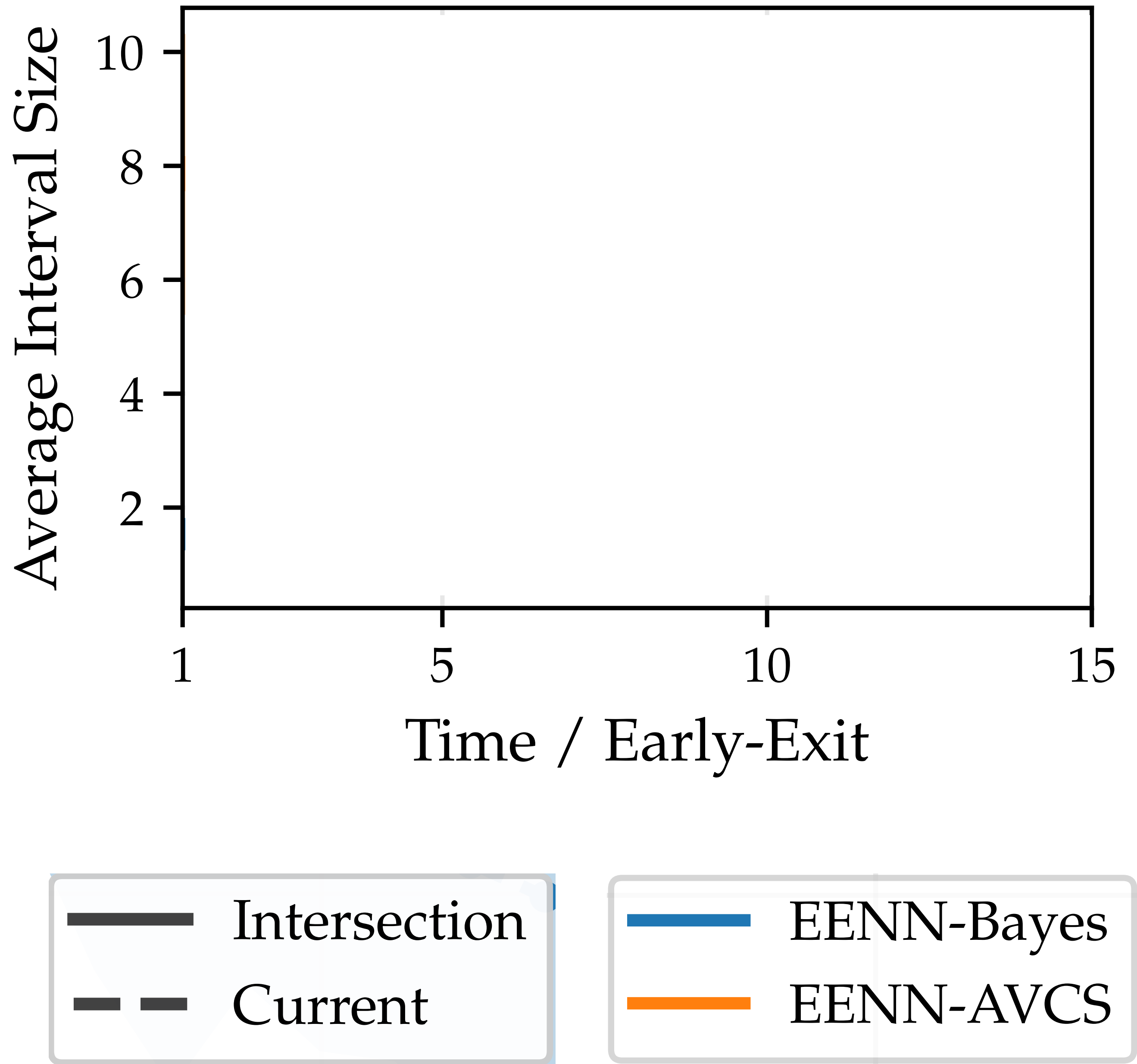


Regression Simulation

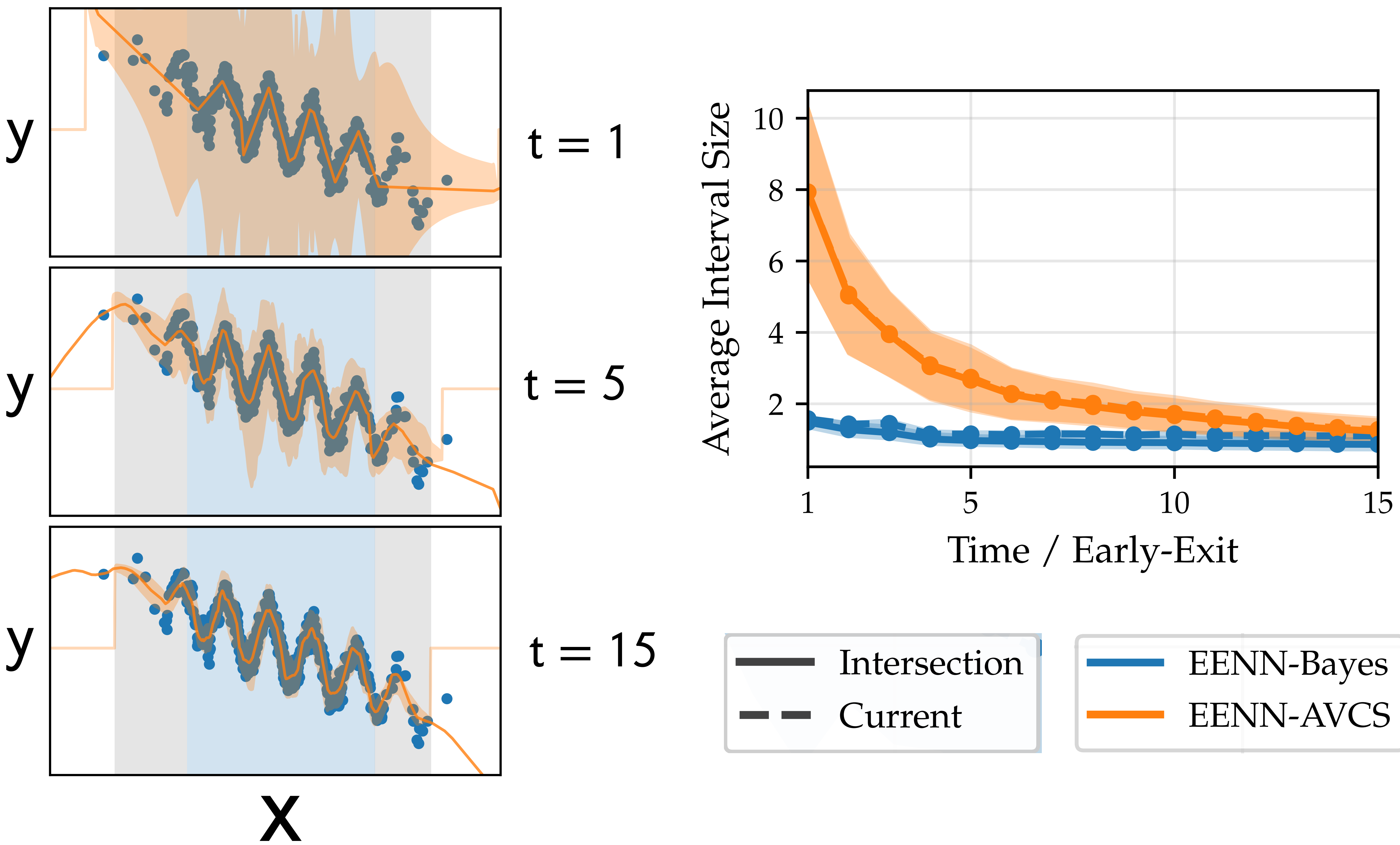


Regression Simulation

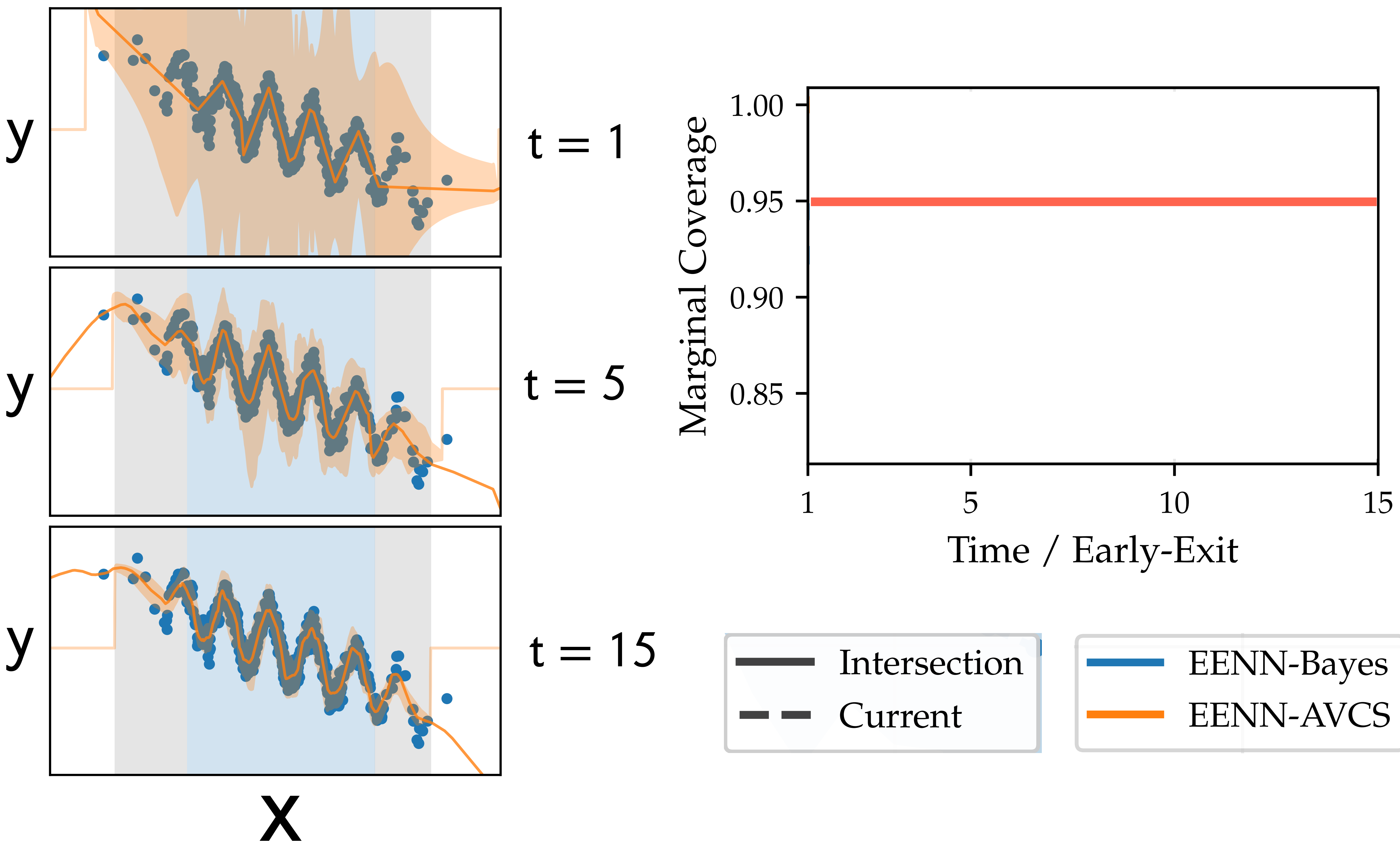




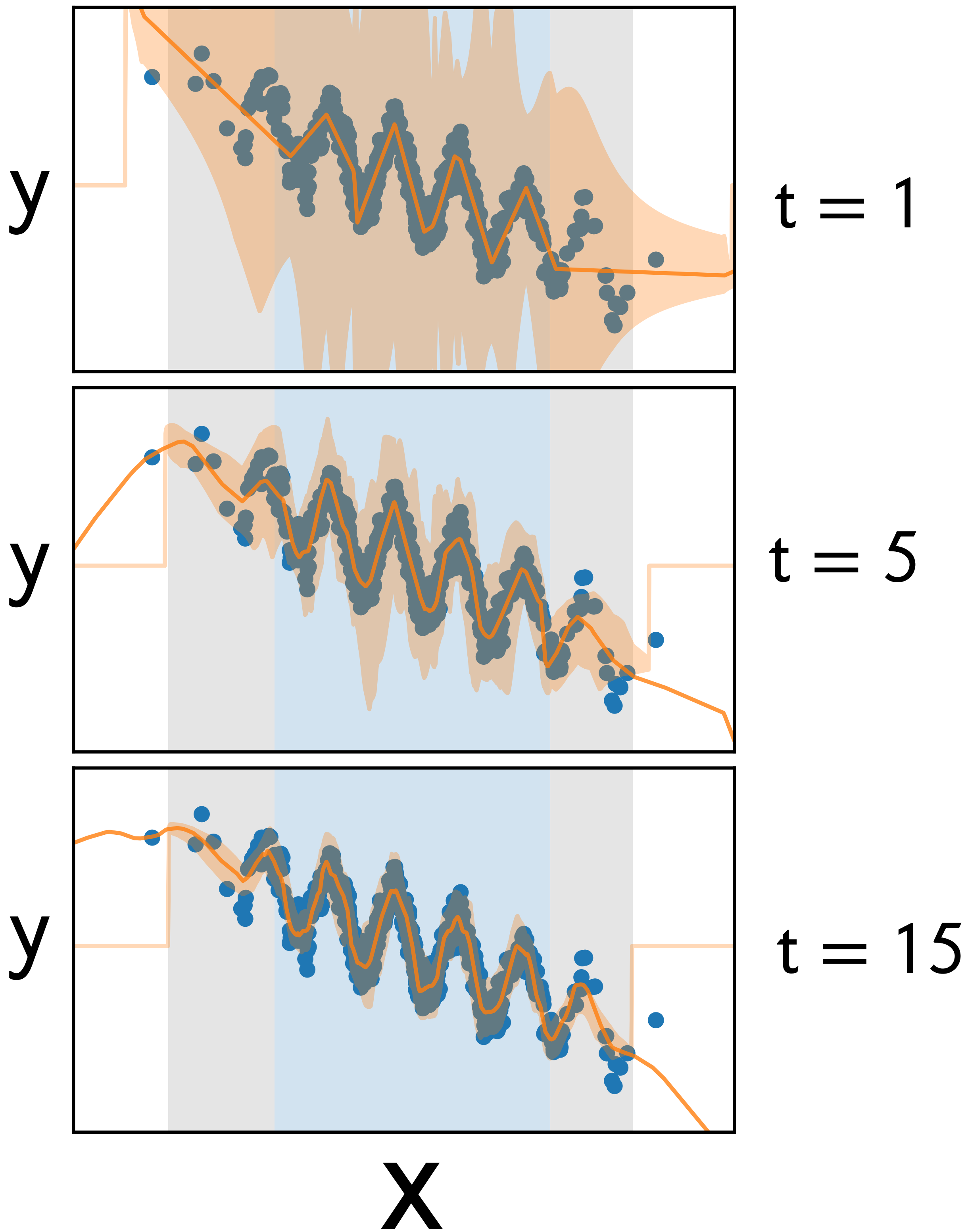
Regression Simulation

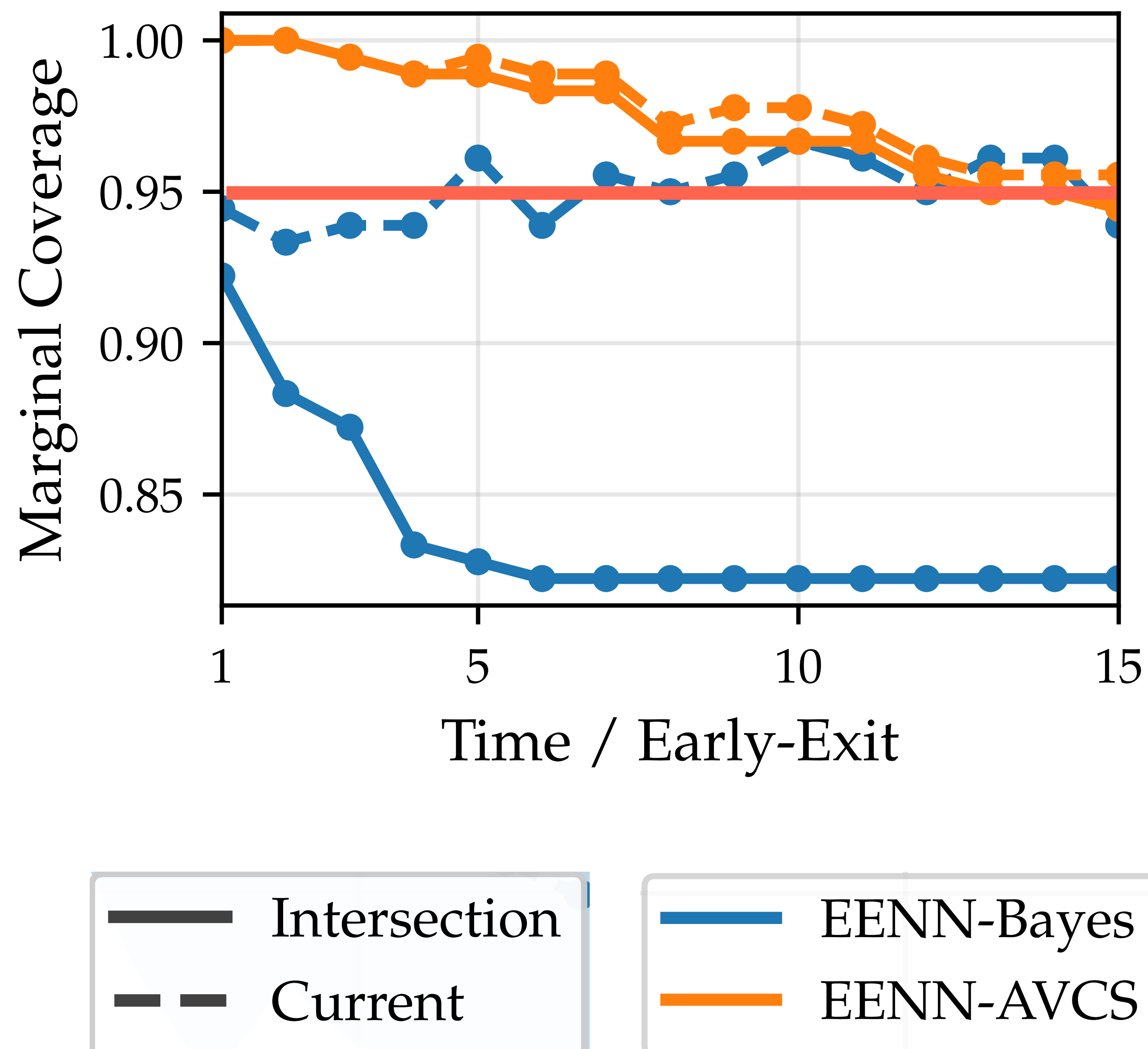


Regression Simulation

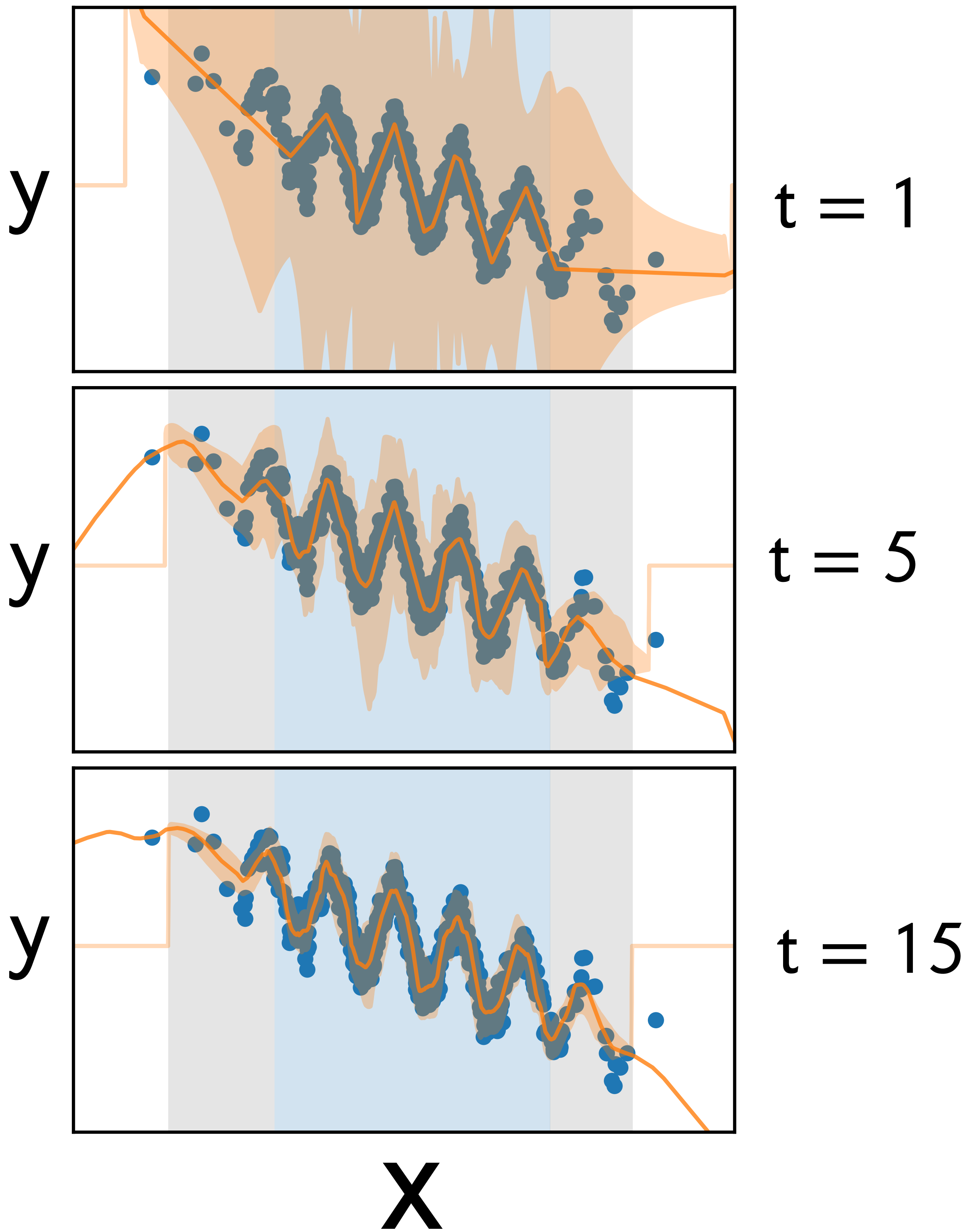


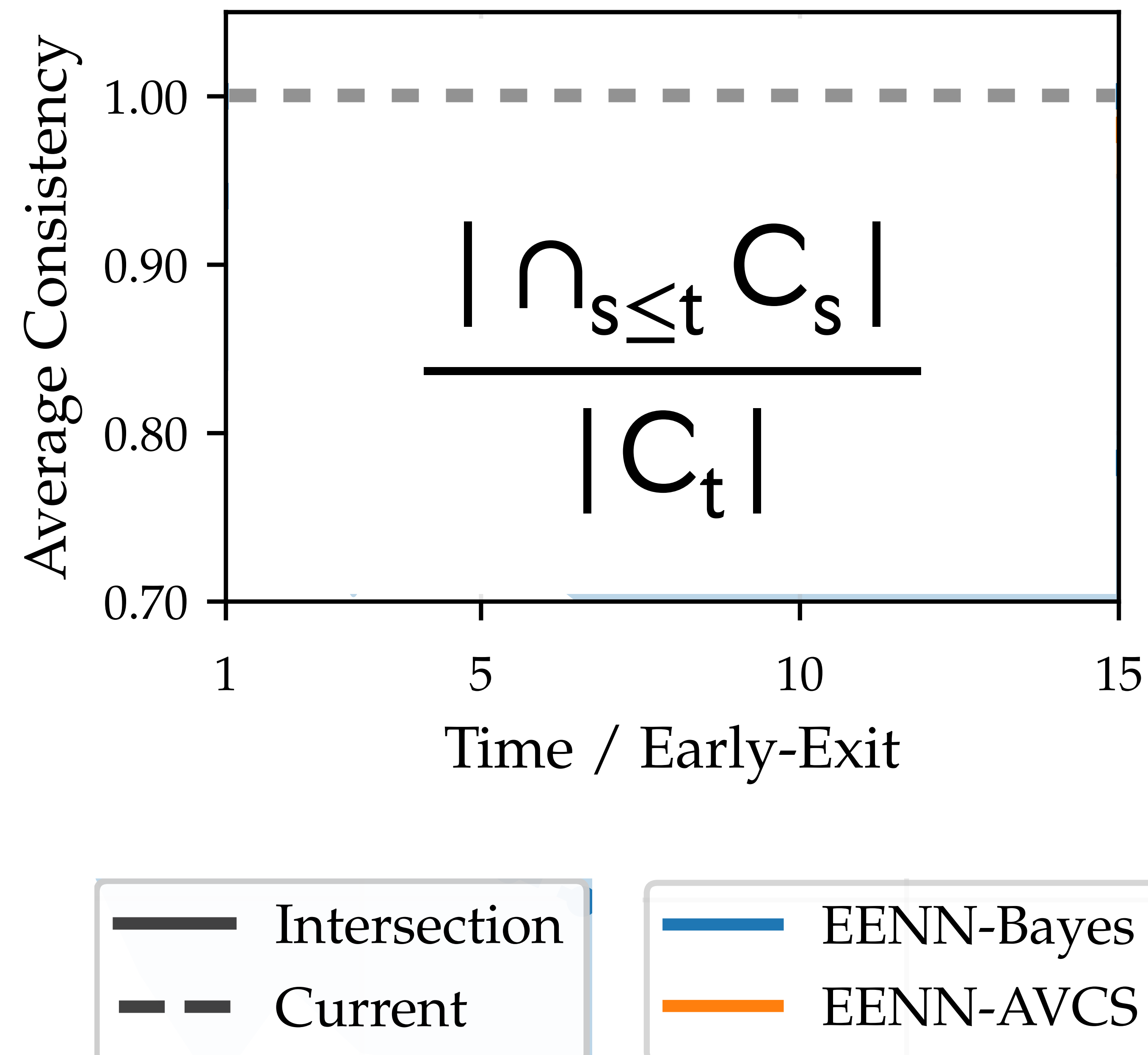
Regression Simulation



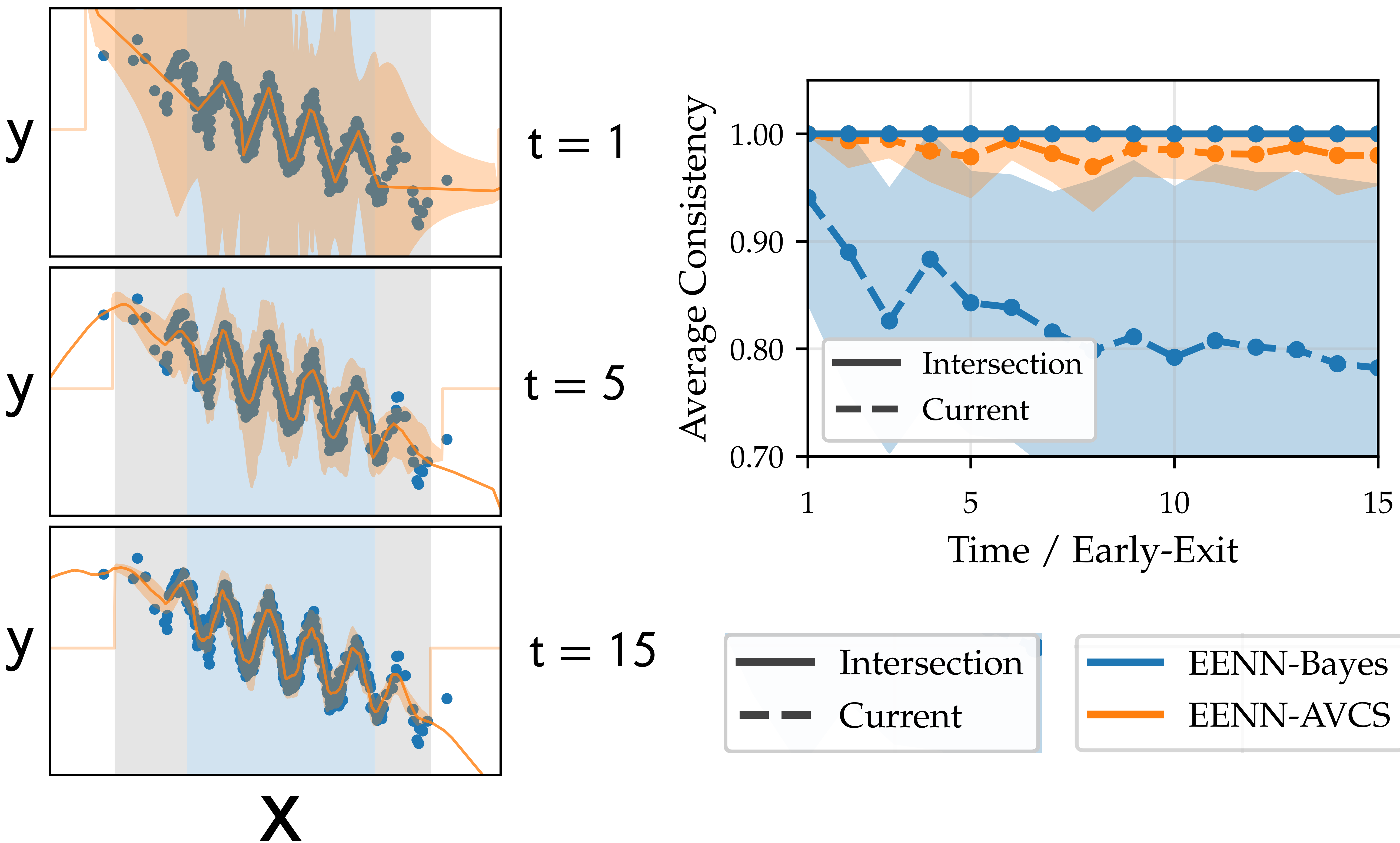


Regression Simulation





Regression Simulation



Summary

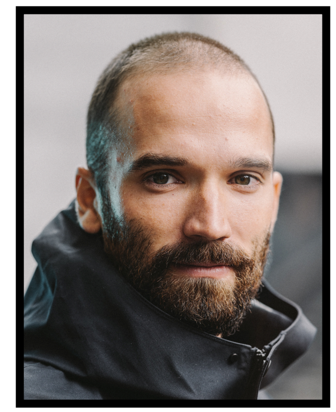
- ⊗ Early-exit neural networks have mostly marginal anytime properties (and overthink)
- ⊗ We give them better conditional monotonicity via a product ensemble.

- ⊗ Also want consistency in predictive uncertainty across exits.
- ⊗ We enforce this with anytime-valid confidence sequences.

Thank you! Questions?

paper

UvA - BOSCH
DELTA LAB



Metod Jazbec

James U. Allingham

Dan Zhang

Patrick Forré

Stephan Mandt