MACHINE LEARNING DEPARTMENT

Carnegie Mellon University

Competence-based Curriculum Learning for Neural Machine Translation

Anthony Platanios

e.a.platanios@cs.cmu.edu

Joint work with Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell

Neural Machine Translation (NMT)

Carnegie Mellon University

- NMT represents the state-of-the-art for many machine translation systems.
- NMT benefits from *end-to-end training with large amounts of data*.
- Large scale NMT systems are often hard to train:
  - Transformers rely on a number of heuristics such as **specialized learning rate schedules** and **large-batch training**. [Popel 2018]

$$\cancel{\mathrm{lr}(t) \triangleq d_{\text{embedding}}^{-0.5} \min\left(t^{-0.5}, t \cdot T_{\text{warmup}}^{-1.5}\right)}$$

Curriculum Learning

Curriculum Learning

	Easy	Medium	Hard
Training Example	*Thank you!*	*Thank you, for being so patient!*	*Thank you, for being so patient today and coming to this talk even though you're probably tired!*

Training Time →

Curriculum Learning

Carnegie Mellon University

	Easy	Medium	Hard
Training Example	*Thank you!*	*Thank you, for being so patient!*	*Thank you, for being so patient today and coming to this talk even though you're probably tired!*
Training Time	→	→	→

Curriculum Learning

Carnegie Mellon University

	Easy	Medium	Hard
Training Example	*Thank you!*	*Thank you, for being so patient!*	*Thank you, for being so patient today and coming to this talk even though you're probably tired!*

Training Time

Avoid getting stuck in bad local optima early on!

- **[Elman 1993]:** Introduced the idea of curriculum learning.
- **[Kocmi 2017, Bojar 2017]:** Empirical evaluation on MT. Final performance is hurt.
- **[Zhang 2018]:** Data binning strategy. The results are highly sensitive on several hyperparameters.

Discrete regimes.

Improvements in training time!

No improvements in performance!

Our Approach

Carnegie Mellon University

We introduce two key concepts:

- **Difficulty:** Represents the difficulty of a training example that may depend on the current state of the learner.

$$\underset{\text{Training Example}}{d(\underset{\downarrow}{s})}$$

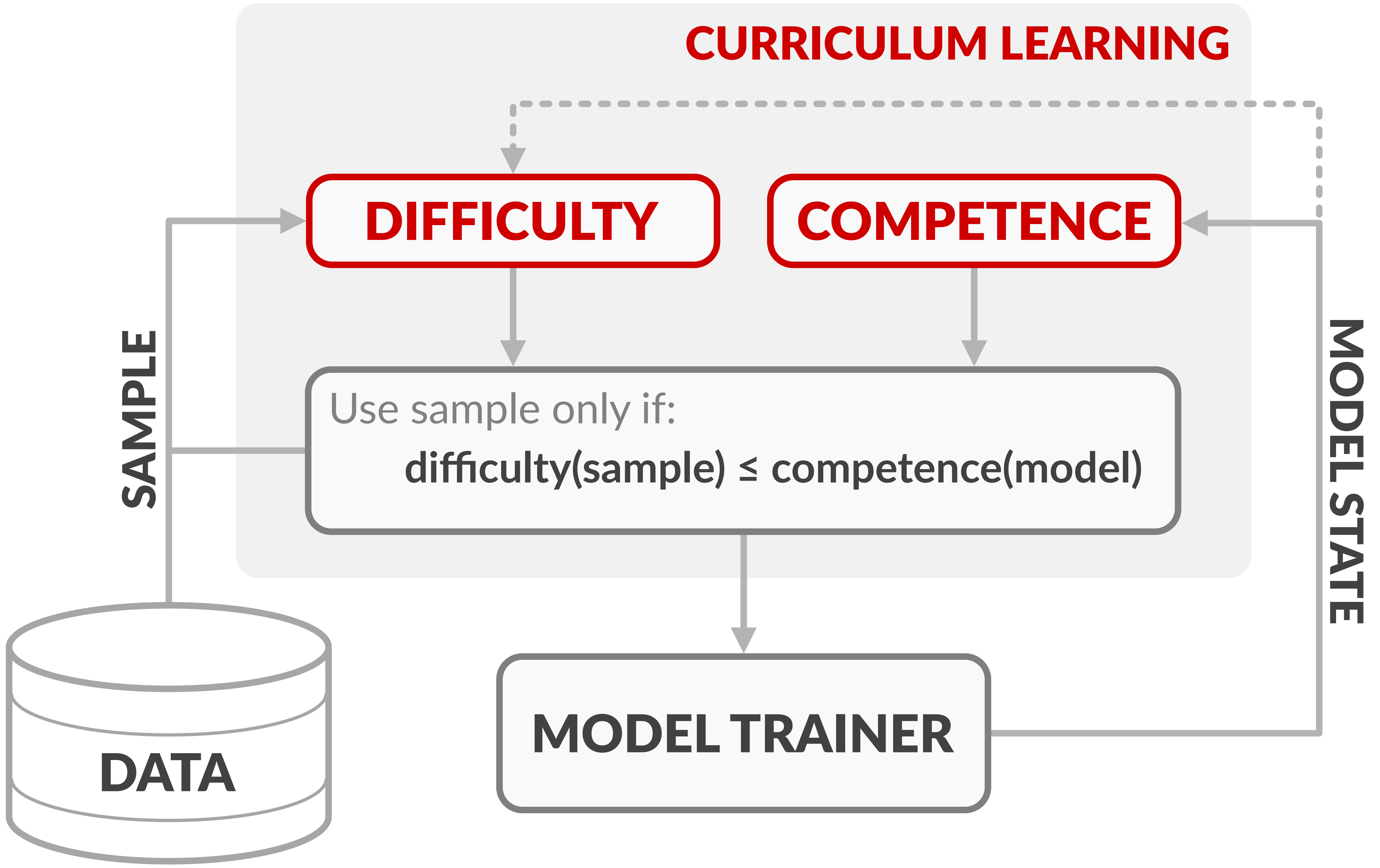
(e.g., sentence length)

- **Competence:** Value between 0 and 1 that represents the progress of a learner during its training and can depend on the learner's state.

$$\underset{\text{Training Step}}{c(\underset{\downarrow}{t})}$$

(e.g., validation set performance)

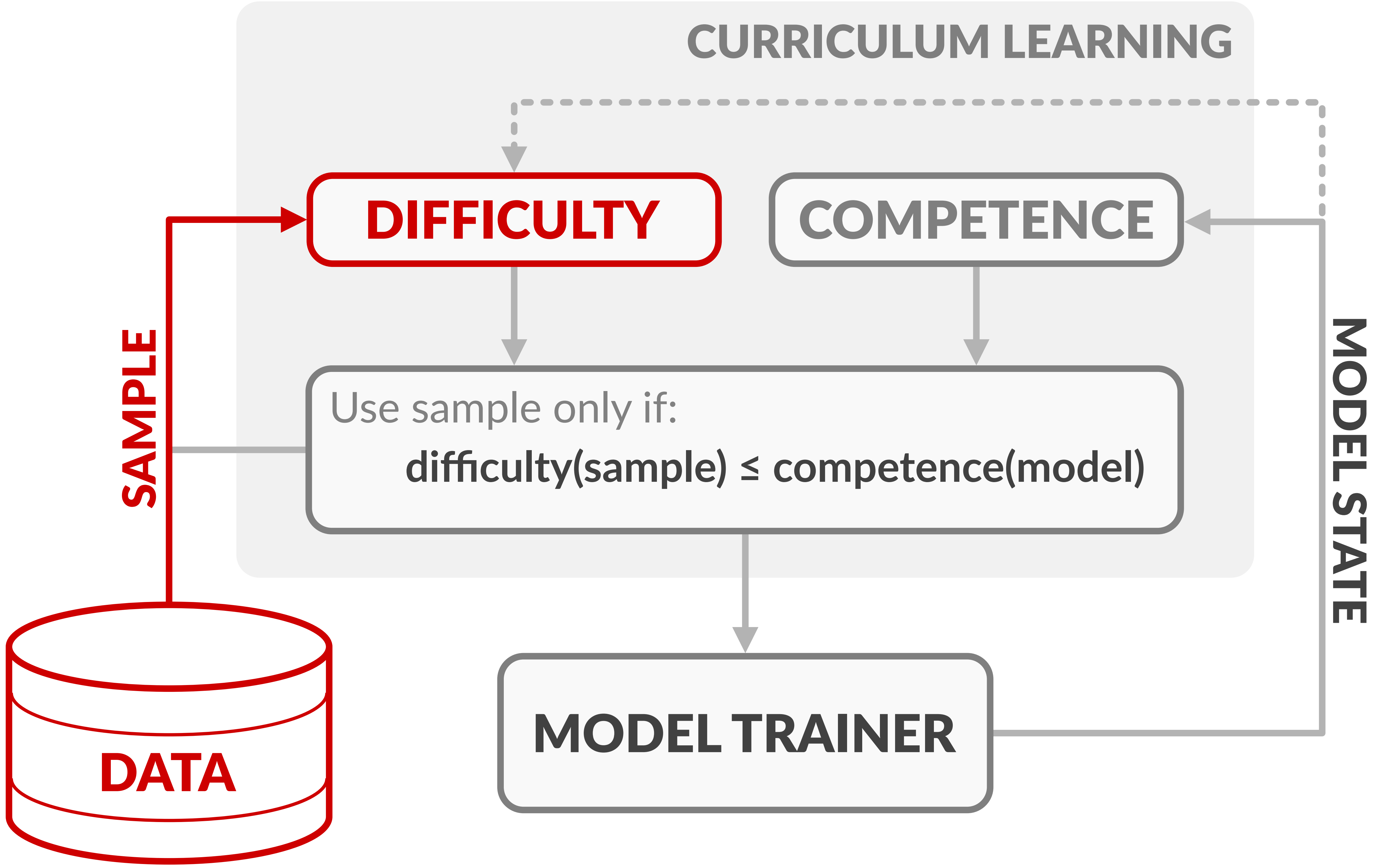
Our Approach



The training examples are ranked according to their difficulty and the learner is only allowed to use the top $c(t)$ portion of them at time t.

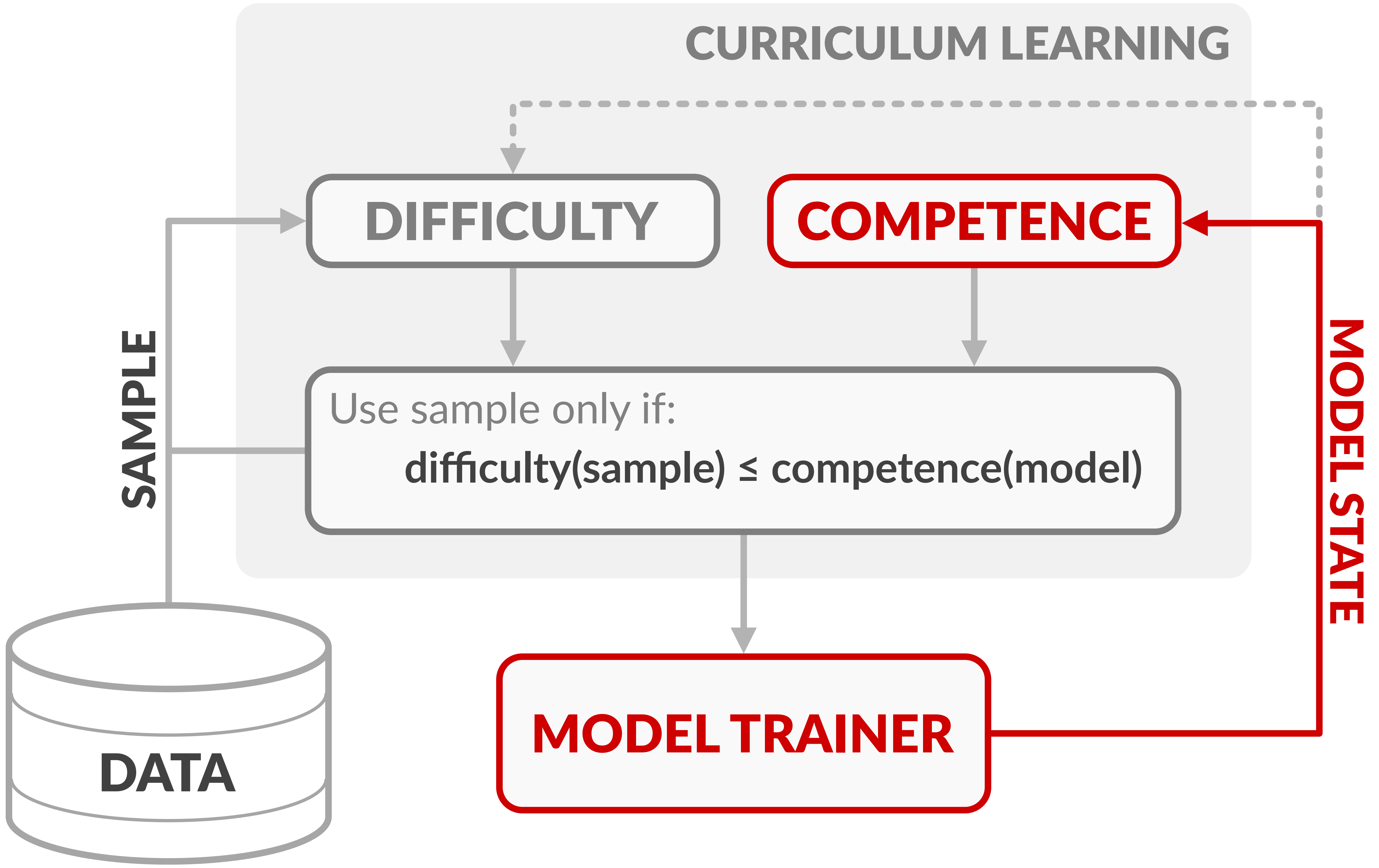
Our Approach

Carnegie Mellon University



The training examples are ranked according to their difficulty and the learner is only allowed to use the top $c(t)$ portion of them at time t.

Our Approach



The training examples are ranked according to their difficulty and the learner is only allowed to use the top $c(t)$ portion of them at time t.

Our Approach — **Algorithm**

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.

Our Approach — **Algorithm**

Carnegie Mellon University

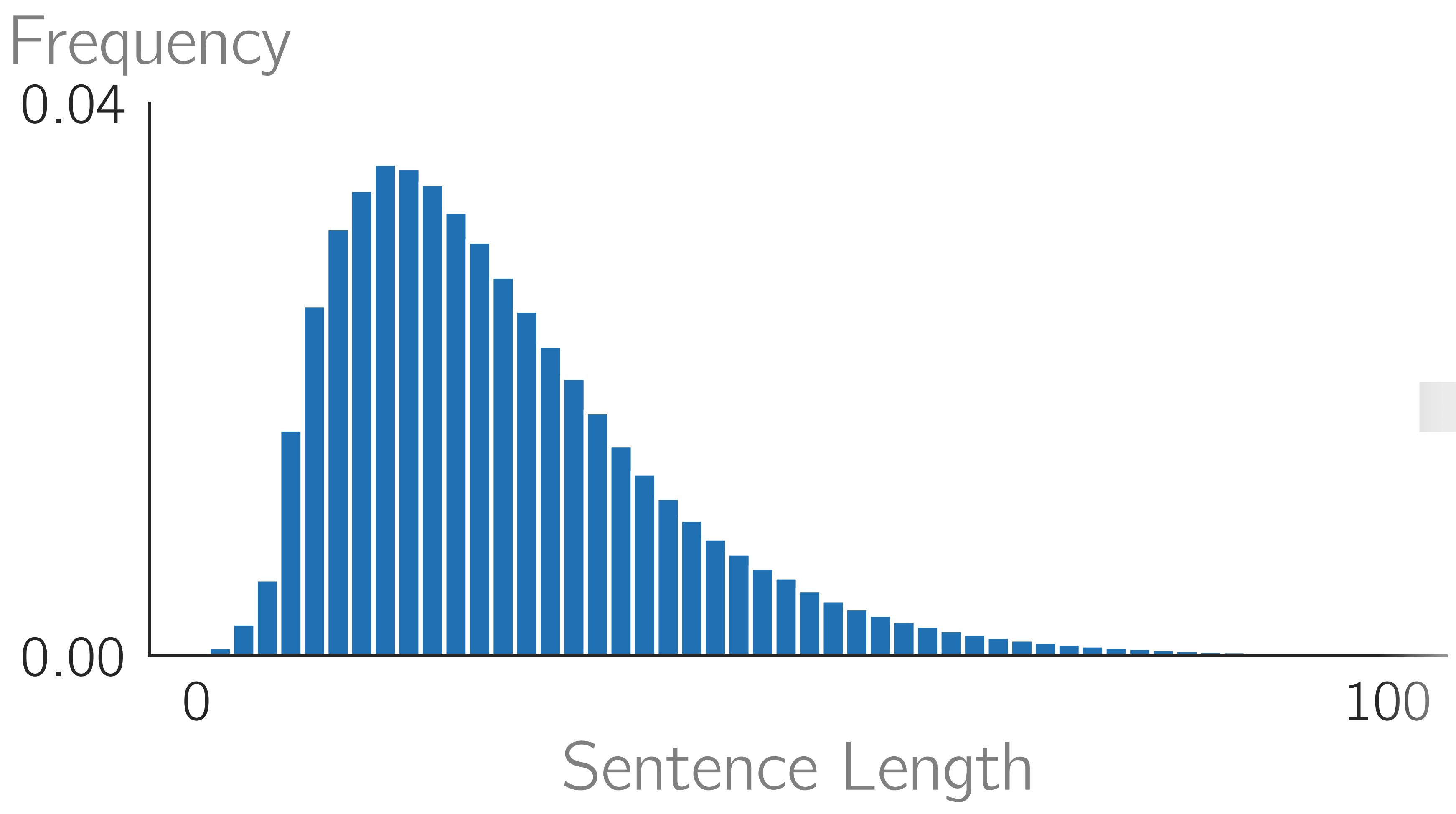
1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.

Our Approach — **Algorithm**

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.

Sentence	**Length**
Thank you very much!	4
Barack Obama loves ...	13
My name is ...	6
What did she say ...	123

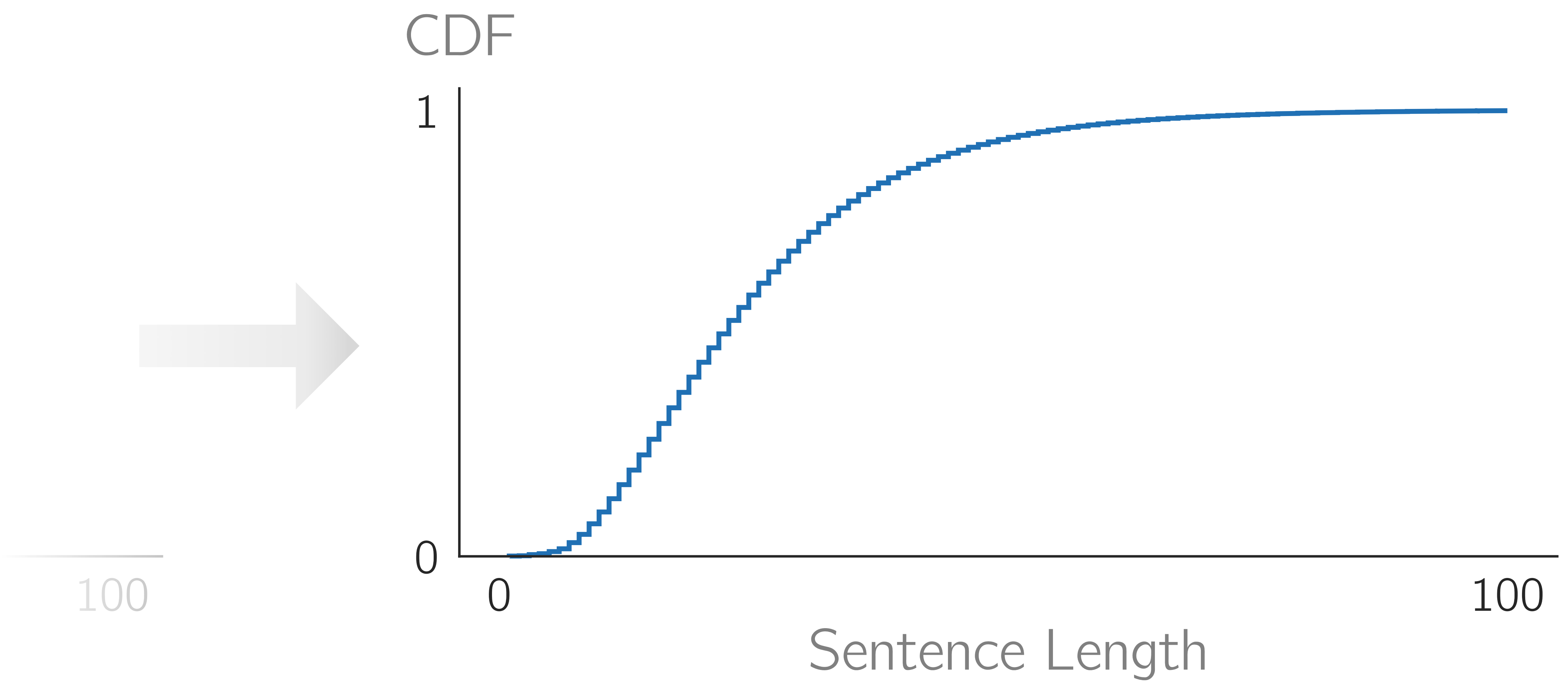
⋮



Our Approach — **Algorithm**

Carnegie Mellon University

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.



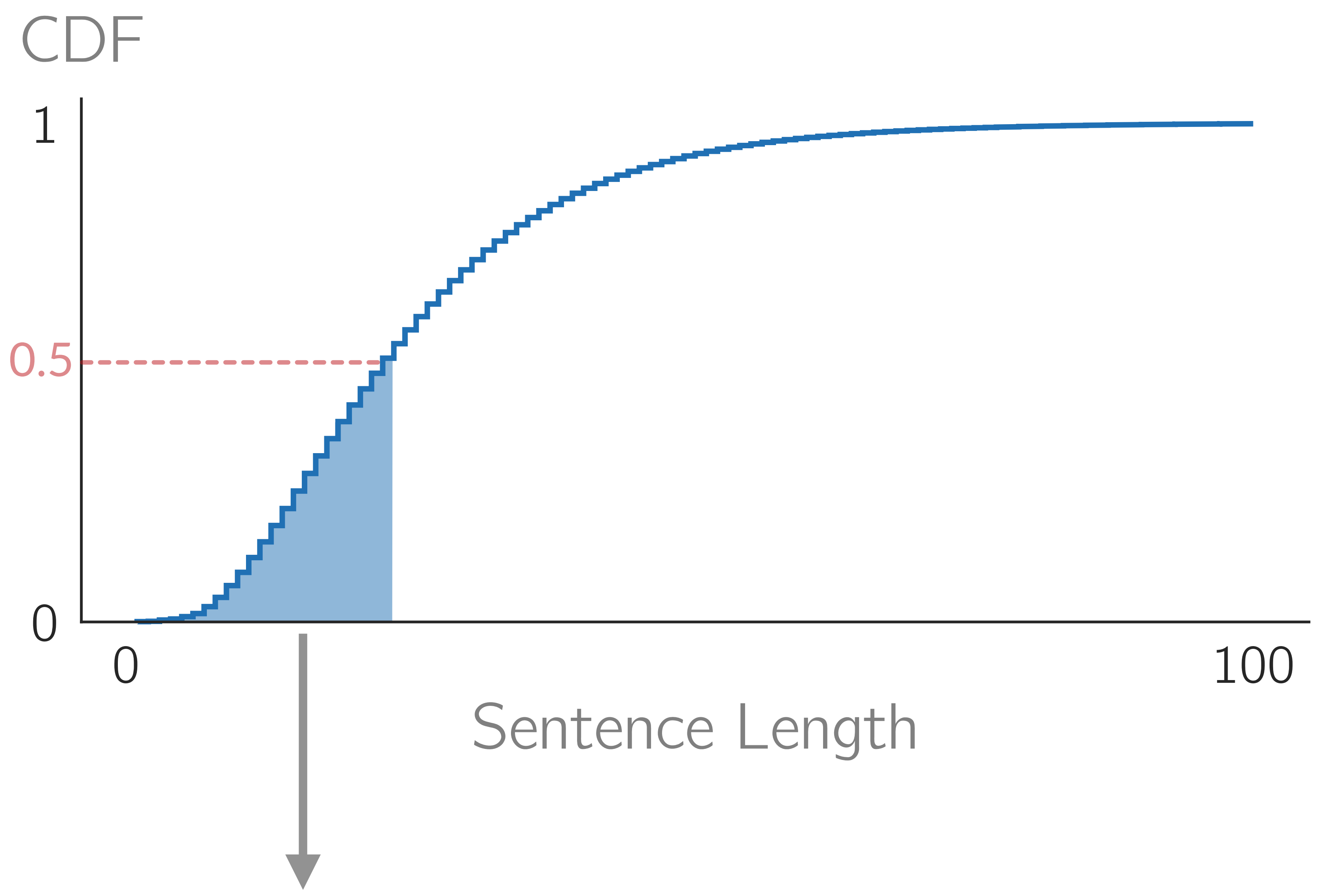
Sentence	Difficulty
Thank you very much!	0.01
Barack Obama loves ...	0.15
My name is ...	0.03
What did she say ...	0.95

⋮

Our Approach — **Algorithm**

Carnegie Mellon University

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.



50% shortest sentences

Sentence	Difficulty
Thank you very much!	0.01
Barack Obama loves ...	0.15
My name is ...	0.03
What did she say ...	0.95

⋮

Our Approach — **Algorithm**

Carnegie Mellon University

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.
3. For training step t = 1, … :
   i. Compute the model competence $c(t)$.

Our Approach — **Algorithm**

Carnegie Mellon University

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.
3. For training step t = 1, … :
   i. Compute the model competence $c(t)$.
   ii. Sample a data batch uniformly from all examples $s_i \in \mathcal{D}$ such that:

$$\bar{d}(s_i) \leq c(t)$$

Our Approach — **Algorithm**

Carnegie Mellon University

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.
3. For training step t = 1, … :
   i. Compute the model competence $c(t)$.
   ii. Sample a data batch uniformly from all examples $s_i \in \mathcal{D}$ such that:
   $$\bar{d}(s_i) \leq c(t)$$
   iii. Invoke the model trainer using the sampled batch.

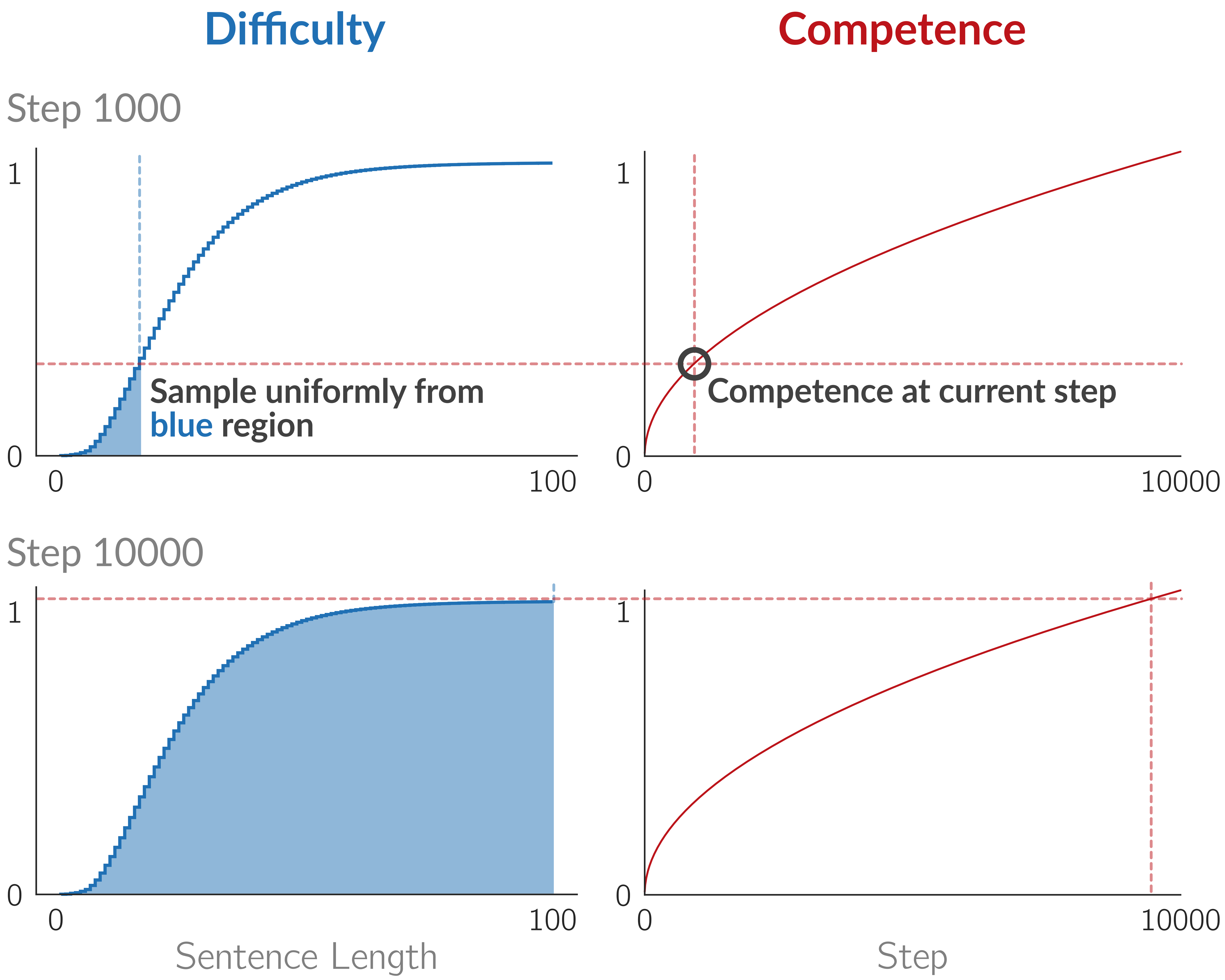
Our Approach — **Algorithm**

1. Compute the difficulty $d(s_i)$ for each $s_i \in \mathcal{D}$.
2. Compute the *cumulative density function* (CDF), $\bar{d}(s_i) \in [0, 1]$, of the difficulties.
3. For training step t = 1, … :
   i. Compute the model competence $c(t)$.
   ii. Sample a data batch uniformly from all examples $s_i \in \mathcal{D}$ such that:
   $$\bar{d}(s_i) \leq c(t)$$
   iii. Invoke the model trainer using the sampled batch.

We are not changing the relative probability of each training example under the input data distribution. We are constraining the domain of that distribution.

Our Approach — **Algorithm**

Carnegie Mellon University



Our Approach — **Difficulty**

Carnegie Mellon University

We denote our training corpus as a **collection of sentences**, $\{s_i\}_{i=1}^{M}$, where each sentence is a **sequence of words**: $s_i = \{w_0^i, \ldots, w_{N_i}^i\}$.

- **Sentence Length:** $$d_{\text{length}}(s_i) \triangleq N_i$$
- **Word Rarity:** $$d_{\text{rarity}}(s_i) \triangleq -\sum_{k=1}^{N_i} \log \hat{p}(w_k^i)$$

$$\hat{p}(w_j) \triangleq \frac{1}{N_{\text{total}}} \sum_{i=1}^{M} \sum_{k=1}^{N_i} \mathbb{I}_{w_k^i = w_j}$$

Our Approach — **Competence**

$c(t)$ → a value in [0, 1] that represents the progress of a learner during its training.

→ proportion of training data the learner is allowed to use at step t.

Our Approach — **Competence**

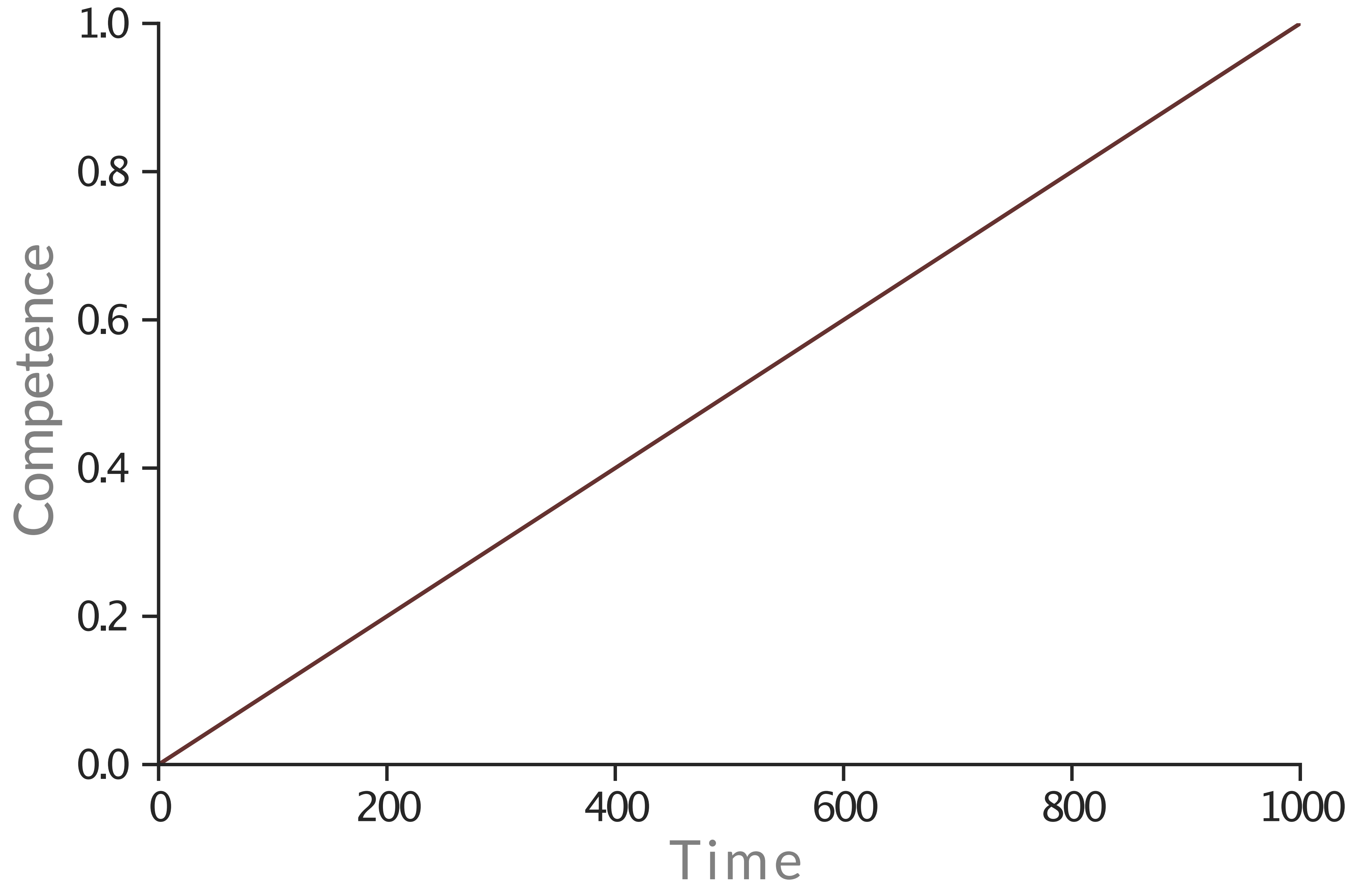
$c(t)$
- a value in [0, 1] that represents the progress of a learner during its training.
- proportion of training data the learner is allowed to use at step t.

Linear Competence

initial competence

$$c_{\text{linear}}(t) \triangleq \min\left(1, t\frac{1-c_0}{T} + c_0\right)$$

time after which the learner is fully competent



Our Approach — **Competence**

$c(t)$ → a value in [0, 1] that represents the progress of a learner during its training.
→ proportion of training data the learner is allowed to use at step t.

Learner-Dependent Competence

E.g., **validation set performance**.

Too Expensive!

Our Approach — **Competence**

$c(t)$ → a value in [0, 1] that represents the progress of a learner during its training.
→ proportion of training data the learner is allowed to use at step t.

Root Competence

Keep the rate in which new examples come in, inversely proportional to the training data size:

$$\frac{dc(t)}{dt} = \frac{P}{c(t)}$$

$$\int c(t)dc(t) = \int P dt \Rightarrow c(t) = \sqrt{2Pt + D}$$

Our Approach — **Competence**

Carnegie Mellon University

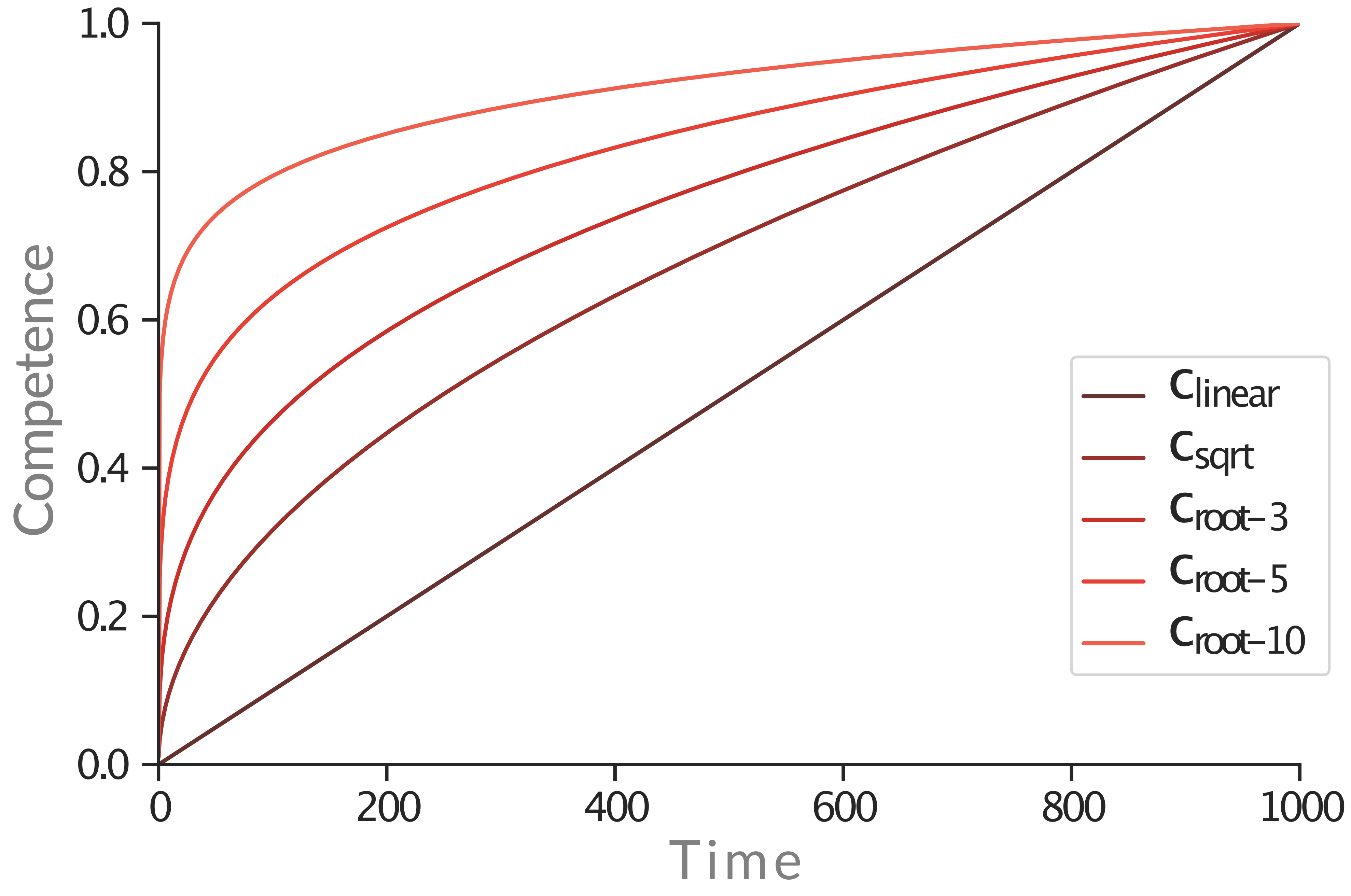
$c(t)$ → a value in [0, 1] that represents the progress of a learner during its training.
→ proportion of training data the learner is allowed to use at step t.

Root Competence

Keep the rate in which new examples come in, inversely proportional to the training data size:

$$c_{\mathrm{sqrt}}(t) \triangleq \min\left(1, \sqrt{t\frac{1-c_0^2}{T} + c_0^2}\right)$$

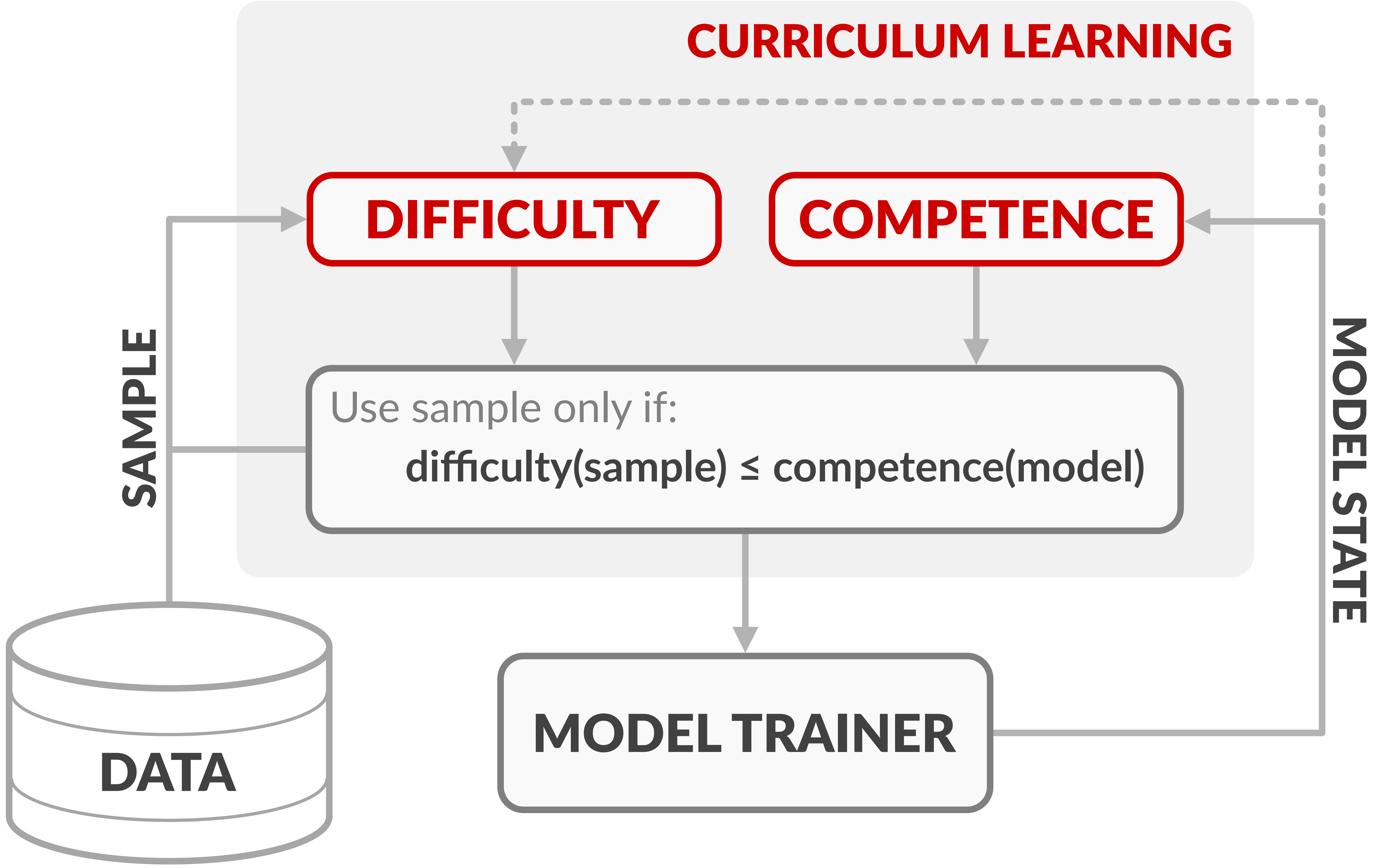
$$c_{\mathrm{linear}}(t) \triangleq \min\left(1, t\frac{1-c_0}{T} + c_0\right)$$



Our Approach

Carnegie Mellon University

DIFFICULTY
- Sentence Length
- Word Rarity



COMPETENCE
- Linear
- Root

The training examples are ranked according to their difficulty and the learner is only allowed to use the top $c(t)$ portion of them at time t.

Experiments — **Datasets**

Dataset	# Train	# Dev	# Test
IWSLT-15 En→Vi	133k	768	1268
IWSLT-16 Fr→En	224k	1080	1133
WMT-16 En→De	4.5m	3003	2999

Experiments — **Setup**

- RNN:
  - 2-layer bidirectional LSTM encoder / 2-layer decoder (4 layers for WMT).
  - 512 hidden units per layer and word embedding size
- Transformer:
  - 6-layer encoder/decoder.
  - 2,048 units for the feed-forward layers and 512 word embedding size.
- AMSGrad optimizer (similar to Adam) with learning rate 0.001
- Label smoothing factor = 0.1
- Batch size = 5,120 tokens (i.e., 256 for sentence length 20)
- Beam width = 10 (using GNMT length normalization)
- BPE vocabulary with 32,000 merge operations

Experiments — **Setup**

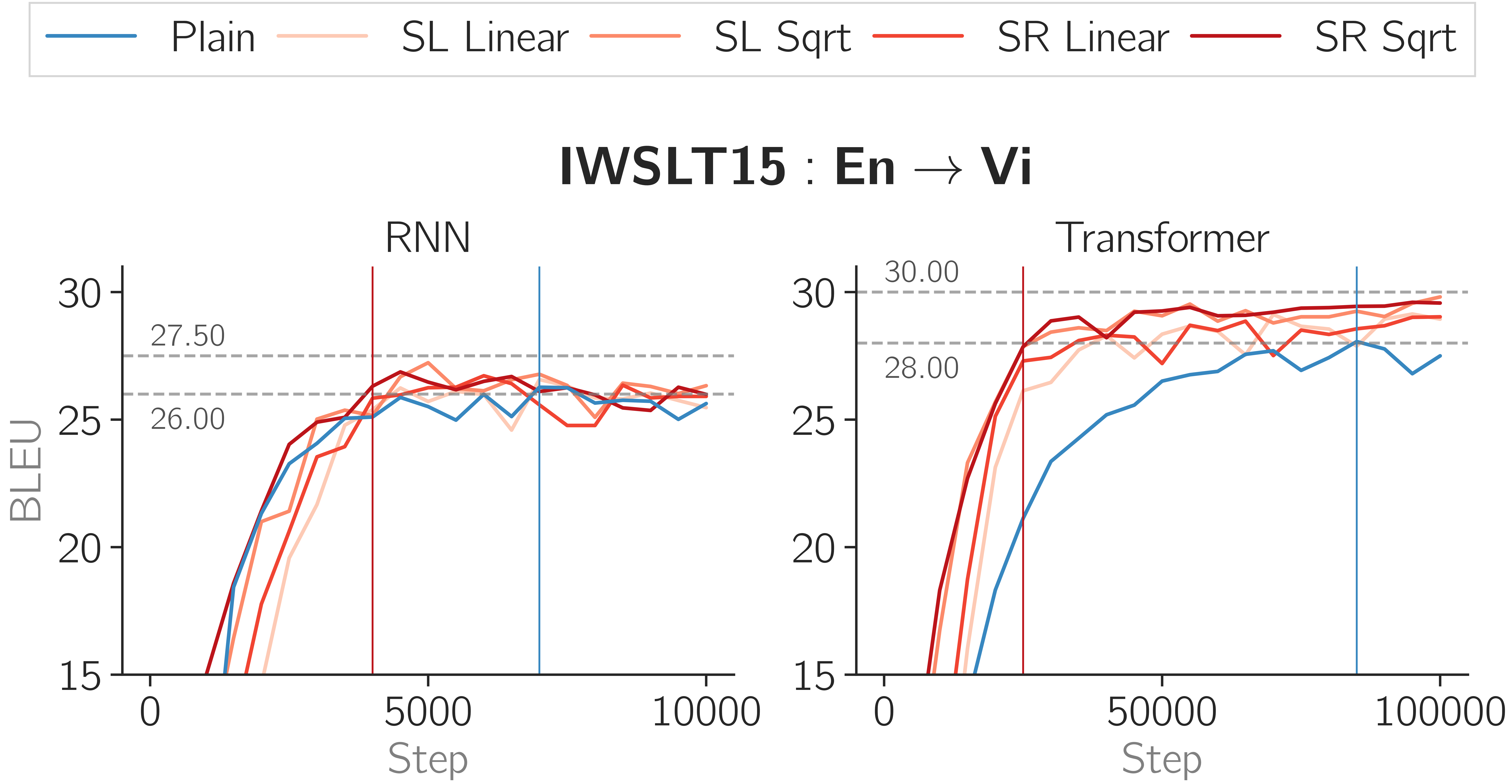
Carnegie Mellon University

Initial Competence: $c_0 = 0.01 \longrightarrow$ All models start training using the 1% easiest training examples.

Curriculum Length: $T \longrightarrow$ We train the baseline model without using any curriculum, and compute the number of training steps it takes to reach ~90% of its final BLEU score.

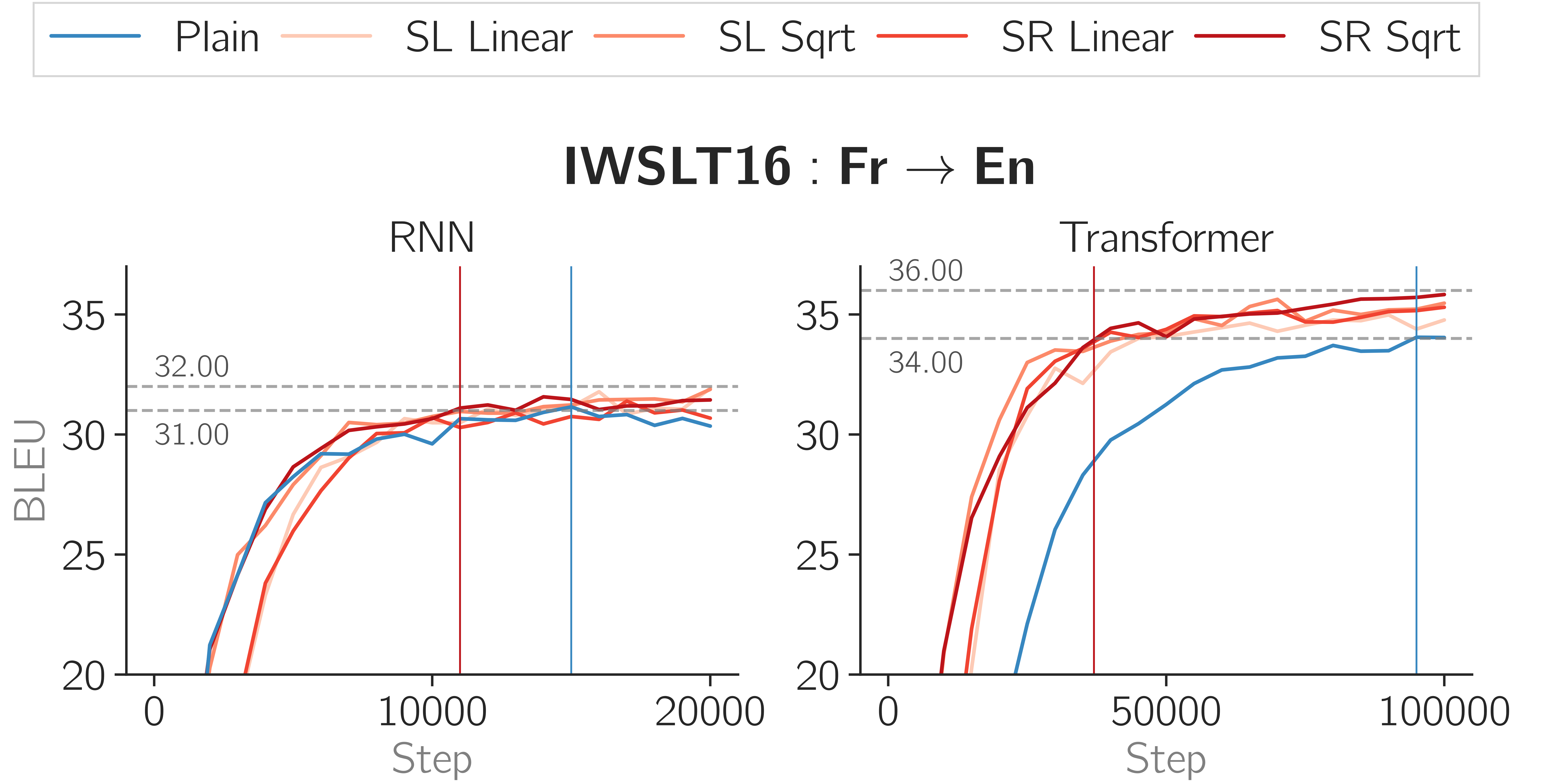
Experiments — **Results**

Carnegie Mellon University

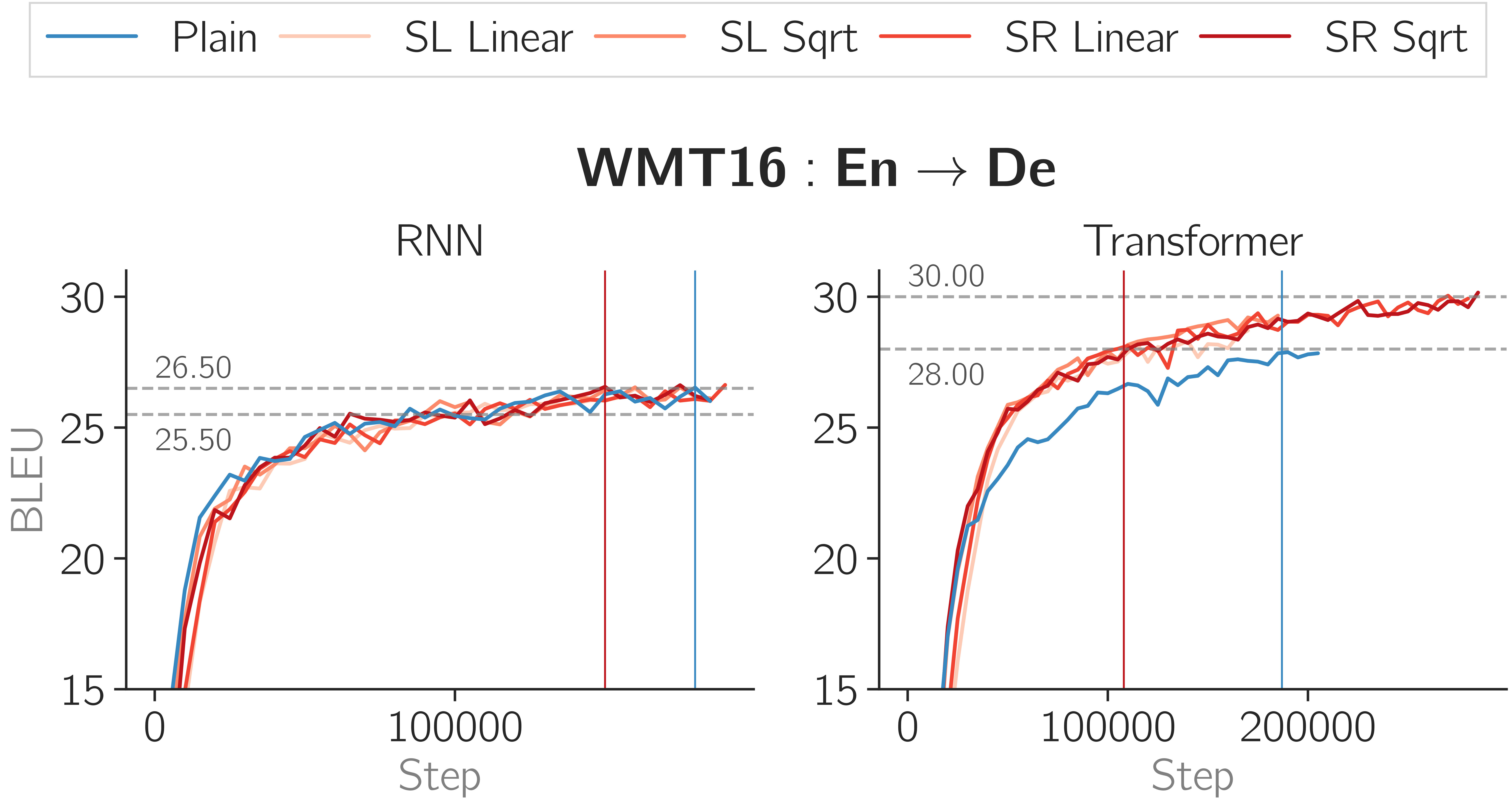


Experiments — **Results**

Carnegie Mellon University



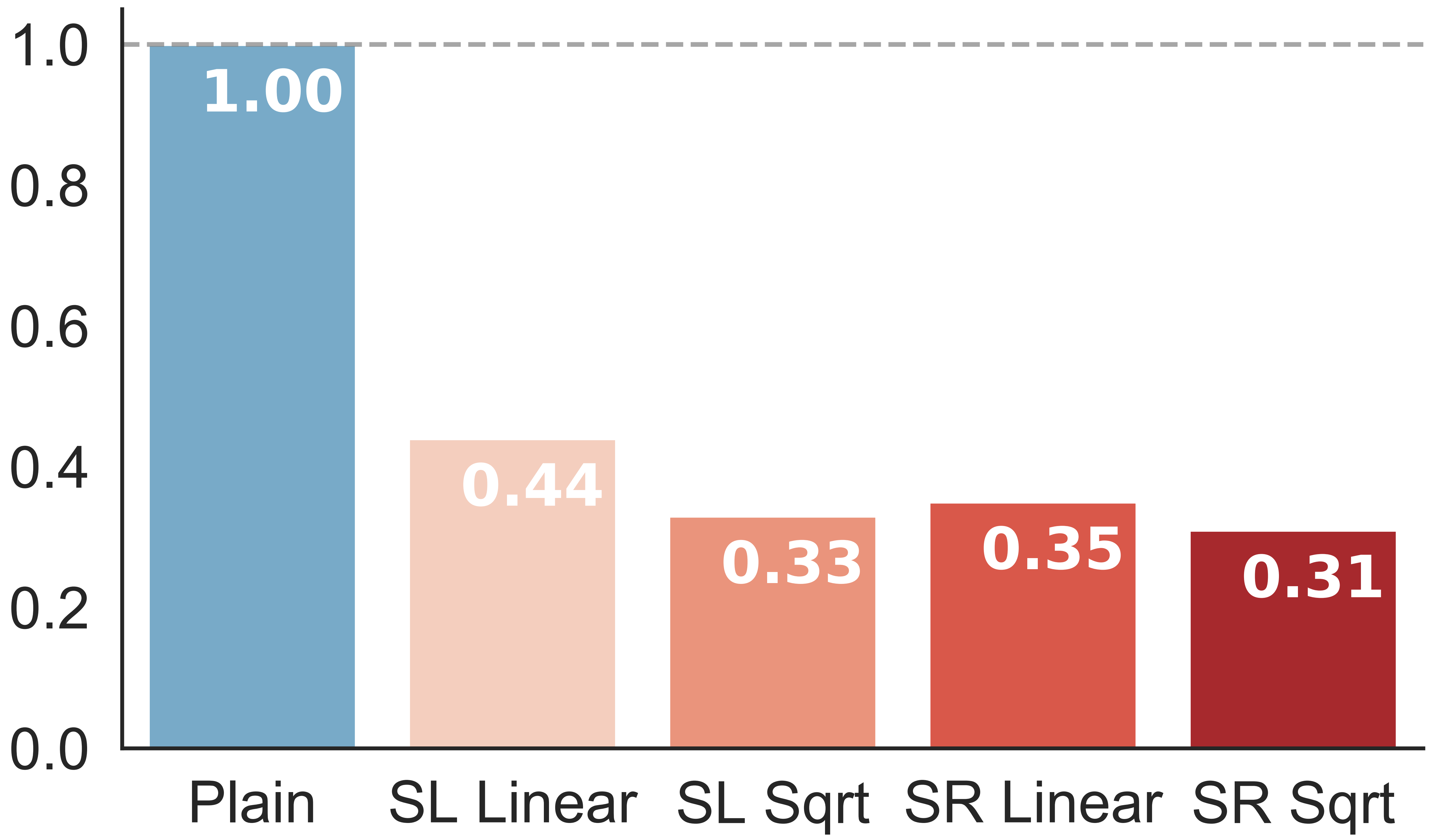
Experiments — **Results**



Experiments — **Results**

Carnegie Mellon University

Relative Time to Baseline Performance



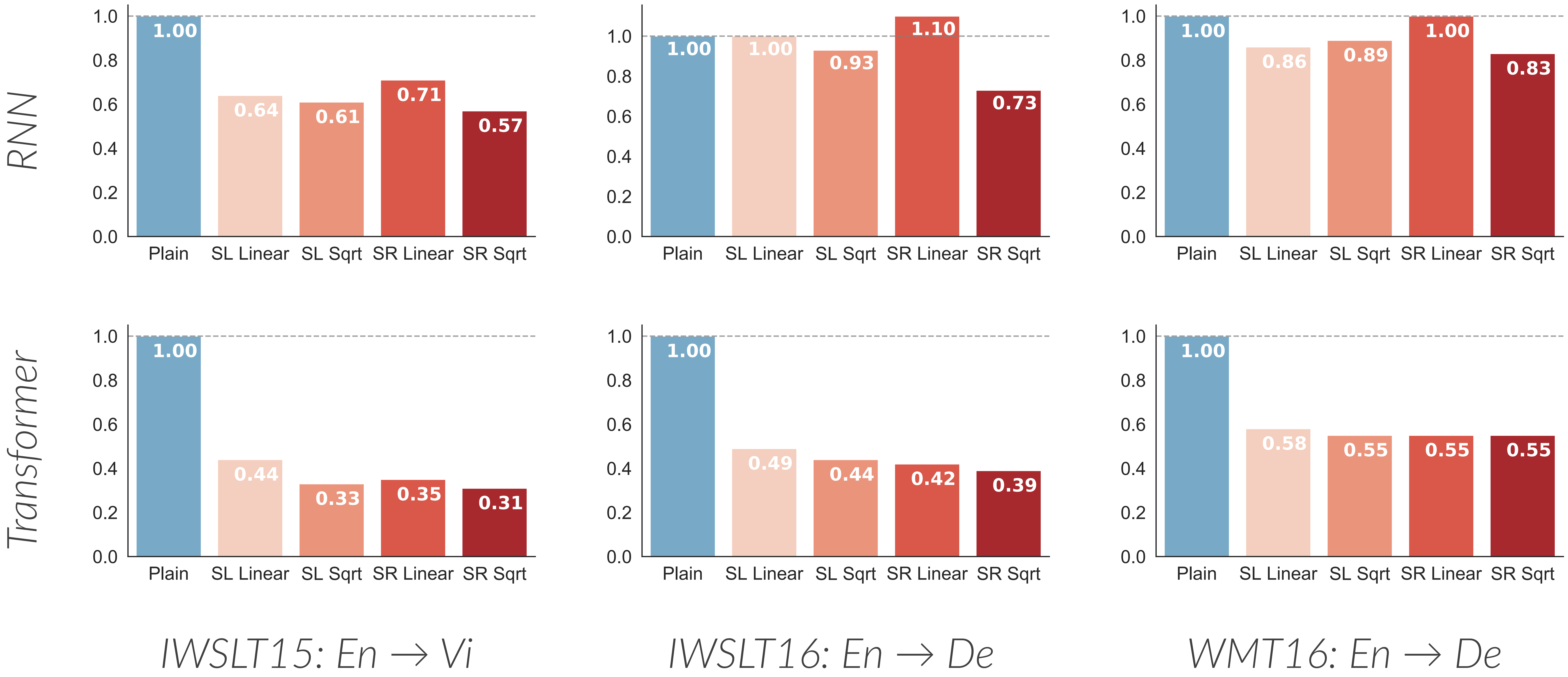
Transformer

IWSLT15: En → Vi

Experiments — **Results**

Carnegie Mellon University

Relative Time to Baseline Performance



Conclusion — **Our Approach**

Carnegie Mellon University

We propose a continuous curriculum learning regime (i.e., no binning), that is:

- **Abstract & Extensible:** Is a generalization of multiple existing approaches.
- **Simple:** Can be applied to existing NMT systems with only a small modification to their training data pipelines.
- **Automatic:** Has no hyper-parameters other than the curriculum length.
- **Efficient:** Reduces training time by *up to 70%*, while improving performance by *up to 2.2 BLEU*.

Also, we perform experiments on both *RNNs* and *Transformers*.

Prior work has not evaluated curriculum learning applied to Transformers.

Thank You!

Questions?

e.a.platanios@cs.cmu.edu