Competence-based Curriculum Learning for Neural Machine Translation

Emmanouil Antonios Platanios e.a.platanios@cs.cmu.edu
Otilia Stretcu ostretcu@cs.cmu.edu
Graham Neubig gneubig@cs.cmu.edu
Barnabas Poczos bapoczos@cs.cmu.edu
Tom M. Mitchell tom.mitchell@cs.cmu.edu

Machine Learning Department — Carnegie Mellon University

Motivation

MACHINE TRANSLATION

Translate from one language to another:

English: *How are you?* → **MT System** → Romanian: *Ce mai faci?*

Large scale neural MT systems are hard to train. For example, Transformers require:

- **Large Batch Training**
- **Specialized Learning Rate Schedules** $\mathrm{lr}(t) \triangleq d_{\text{embedding}}^{-0.5} \min\left(t^{-0.5}, t \cdot T_{\text{warmup}}^{-1.5}\right)$

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 →

Previous curriculum learning approaches for NMT:

- **Discrete Regimes**
- **Improvements in Training Time**
- **No Improvements in Performance**

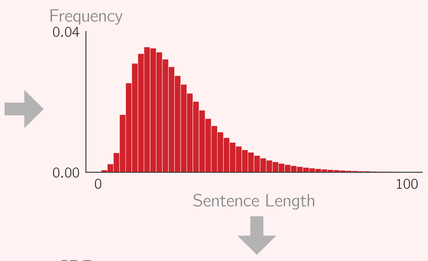
Proposed Approach

DIFFICULTY

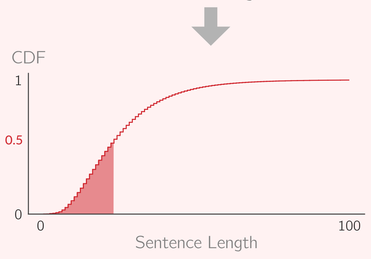
Represents the difficulty of a training example that may depend on the state of the learner:

$d(s)$ *(e.g., sentence length)* → *Training Example*

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



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



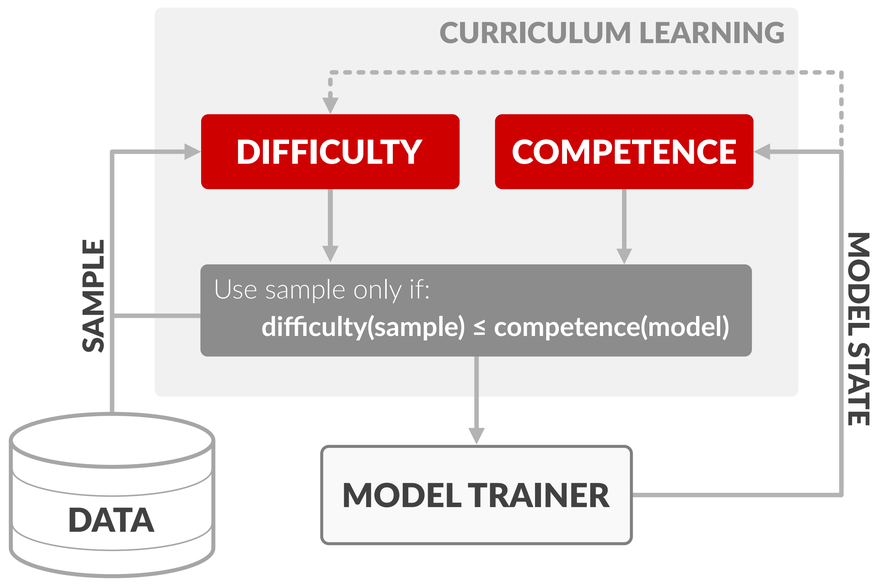
Sentence Length

$d_{\text{length}}(s_i) \triangleq N_i$ → *#words in sentence*

Sentence Rarity

$$d_{\text{rarity}}(s_i) \triangleq -\sum_{k=1}^{N_i} \log \hat{p}(w_k^i)$$

s_i: *sentence*; $\hat{p}(w_k^i)$: *word frequency*



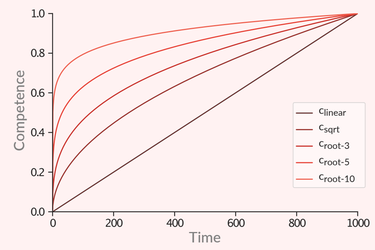
ALGORITHM

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

COMPETENCE

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

$c(t)$ → *Training Step*



(e.g., validation set performance)

Linear Competence

New training examples are constantly being introduced during the training process with a constant rate $r = (1 - c_0)/T$, as a proportion of the total number of available training examples:

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

T: *time after which the learner is fully competent*; c_0: *initial learning rate*

Square Root Competence

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

$$\frac{dc(t)}{dt} = \frac{P}{c(t)} \rightarrow c_{\text{sqrt}}(t) \triangleq \min\left(1, \sqrt{t\frac{1 - c_0^2}{T} + c_0^2}\right)$$

Experiments

DATASETS

- IWSLT-15 (En→Vi)
- IWSLT-16 (Fr→En)
- WMT-16 (En→De)

MODELS

- RNN:
  - Bidirectional LSTM encoder
  - LSTM decoder
  - BPE vocabulary
- Transformer:
  - Base model of
  - BPE vocabulary Vaswani et al.

PARAMETERS

- **Initial Competence:** 0.01
- **Curriculum Length:** We train the baseline model without any curriculum, and compute the number of training steps it takes to reach ~90% of its final BLEU score.

CODE

Scala MT library to reproduce experiments:
https://github.com/eaplatanios/symphony-mt

TensorFlow Scala used for our experiments:
https://github.com/eaplatanios/tensorflow_scala

TRANSLATION BLEU SCORE

RELATIVE TIME TO BASELINE PERFORMANCE