Universal pre-training

by iterated random computation

Peter Bloem 30 July 2025 p@peterbloem.nl @pbloem@sigmoid.social

VU VRIJE UNIVERSITEIT AMSTERDAM

---

talk structure

part one: intuition

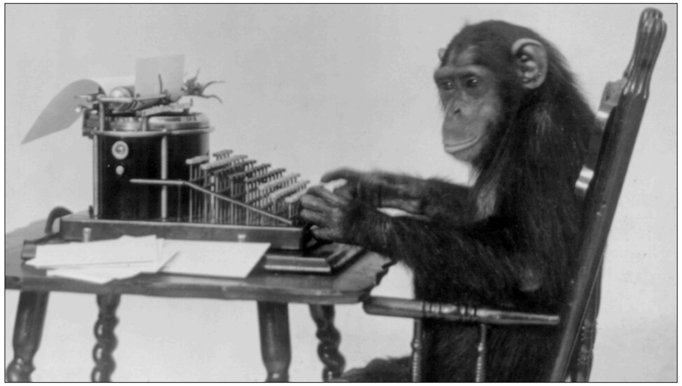
part two: theory

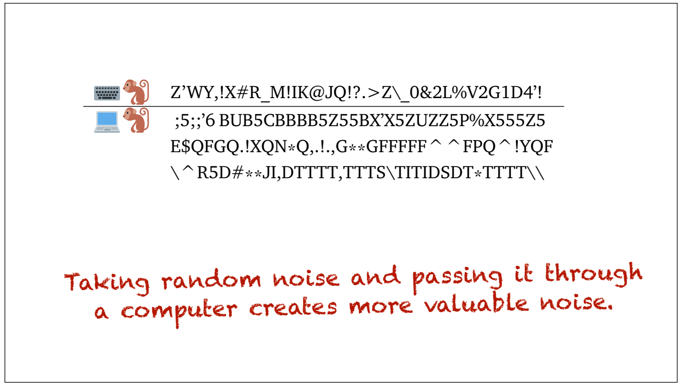
part three: practice

---

intuition

Monkey typewriter/computer





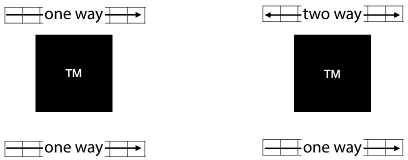
in theory

NB: We're moving away from approximation land for this part. We're happy with uncomputable functions for the time being.

monotone vs prefix

monotone | prefix

output tape



input tape

distributions on infinite sequences
sequential prediction $p(x_i \mid x_{<i})$

distributions on finite sequences
"object" prediction $p(x)$

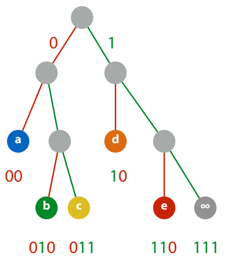
preliminaries

$$U(\bar{\imath}q) = T_i(q)$$ <- prefix TM

$$K(x) = \min\{\,|p| : U(p) = x\}$$

Finite strings only, no prediction

TMs as probability distributions / semimeasures



Feed a (prefix-free) TM random bits until it produces an output.

$$p(x) = \sum_{p : TM(p) = x} 2^{-|p|}$$

If your TM is a universal turing machine, then L1 and L2 correspond (up to a constant). But for other TMs, they may disagree arbitrarily much.

Taken over all TMs the set of probability distributions we can define this way (or more accurately, probability semimeasures) corresponds to the lower semicomputable semimeasures.

---

class-bounded Kolmogorov complexity $K_C(x)$

see also my presentation at the previous AIT symposium: peterbloem.nl/publications/safe-approximation

- Pick a subset C of prefix TMs corresponding to some model class.
  e.g. Markov models, VAEs, Diffusion models, all polynomial-time TMs
- Assign some prior probability p(c) to each c in C.
- Compute the mixture probability $m_C(x)$ of x under C with prior p.

$$m_C(x) = \sum_{c \in C} p(c) p_c(x)$$

- The class-bounded Kolmogorov complexity is -log $m_C(x)$. It is
  computable if every c in C is well-behaved ("sufficient")
  a *safe approximation* of K(x) if x was generated by a model in C.

---

domination

p(x) dominates q(x) if for all x

$$-\log p(x) + c < -\log q(x)$$
$$p(x) \times c > q(x)$$

m(x) dominates any computable distribution

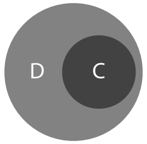
$m_C(x)$ dominates any distribution in C

Lemma 4.1. *For model classes* C, D *if* D *contains a turing machine* $u((i, x)) = T_i(x)$ *with* i *enumerating* C *and* $(,)$ *a prefix-free pairing function, then* m_D *dominates* C.

Proof.

$$m_D(x) = \sum_{d \in D, r \in R} p(d) 2^{-|r|} \text{ with } R(r \mid d(r) = x)$$
$$\geqslant \sum_r p(u) 2^{-|r|} = p(u) m_C(x)$$

□



three results

- Solomonoff induction also works with prefix TMs *and* under class-bounds.
- "Enriching" noise can be iterated to further enrich it
  If we do this carefully, we build towards m(x)
- Sampling random LSTMs and iterating approximates m(x) in the limit

class-bounded, prefix-free Solomonoff induction

First, for a probability p in $\mathbb{B}$ we write the conditional probability of seeing a *prefix* x continue with the bit b as $p(b \mid x)$. This is defined as

$$p(b \mid x) = \frac{p(xb\_)}{p(x\_)}$$

<- set of all strings with prefix x

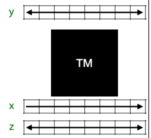
for distributions p and q, define

$$D_n = \sum_{|x|=n} p(x\_) \text{KL}(p(b \mid x), q(b \mid x))$$

Theorem 4.2 (Adapted from Theorem 5.2.1 in [24])**.** *If* q *dominates* p*, then* $\sum_{n=1}^{\infty} D_n$ *is bounded.*

enriching noise by iterating random computation

- Use TMs with a (two-way) conditional input tape $p_C(x \mid z)$
- $m_C^0(x)$: uniform random distribution
  with some prior on string length |x|
- $m_C^{n+1}(x)$: distribution obtained by:
  - sampling z from $m_C^{n+1}(x)$
  - sampling c from p(C)
    the class prior
  - sampling x from $p_C(x \mid z)$



use a random computation to enrich some simple noise, then repeat

---

enriching noise by iterating random computation

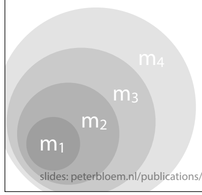
Theorem 4.3. *Let* $i \in C$. *Then* m_C^{n+1} *dominates* m_C^n.

i : identity $p_i(x|x) = 1$

Proof.

$$
\begin{aligned}
m_C^{n+1}(x) &= \sum_{u \in \mathbb{B}, c \in C} m_C^n(u)\, p(c)\, p_c(x \mid u) \\
&= \sum_{u,c} \left(\sum_{u',c'} m_C^{n-1}(u') p(c') p_{c'}(u \mid u') \right) p(c)\, p_c(x \mid u) \\
&\geqslant \sum_{u,c} \left(\sum_{u'} m_C^{n-1}(u') p(i) p_i(u \mid u') \right) p(c)\, p_c(x \mid u) \\
&= \sum_{u,c} m_C^{n-1}(u)\, p(i)\, p(c)\, p_c(x \mid u) \\
&= p(i)\, m_C^n(x)
\end{aligned}
$$

$u = u'$



□

---

using LSTMs

Lemma 4.4. *Let* $p(f)$ *be the probability that an LSTM with* n *parameters, initialized from a given non-degenerate Gaussian over its parameters computes the function* f. *If there exists one such initialization, then there exists some* $\epsilon > 0$ *such that* $p(f) > \epsilon$

Theorem 4.5. *Let* $m_{UTM}^n(x)$ *be the distribution defined by running a universal Turing machine (UTM) on a random input for* n *steps, and observing the output* x. *If* m_C *dominates* C_{LSTM} *and* $r, s \in C$, *then* m_C^{n+2} *dominates* m_{UTM}^n.

proof idea: find the LSTM that simulates one step of the UTM

r, s: simple utility functions

in practice

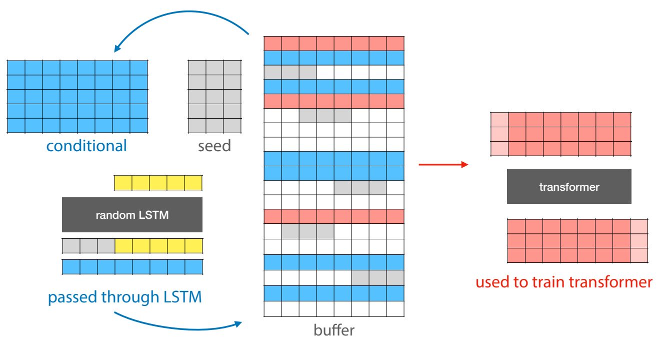
What has changed since 2014, 2015?

The basic idea

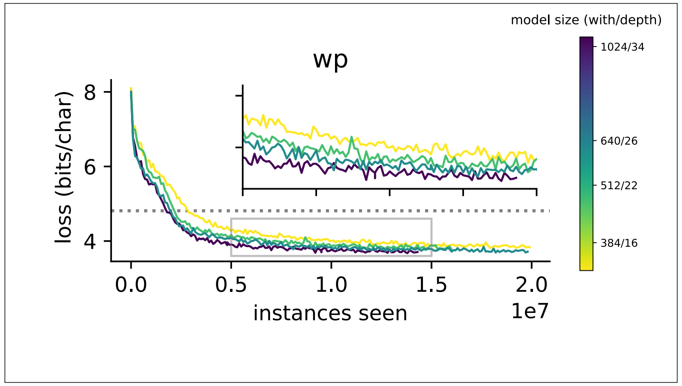
- Put random noise through a random LSTM
- iterate n times with different LSTMs
- Pre-train an autoregressive transformer on this noise
- Check the zero-shot performance on Wikipedia test (and other data)

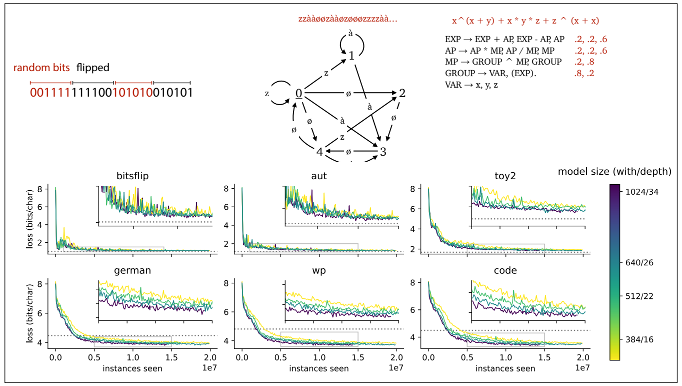
problem: if we batch we get n samples from one LSTM
problem: cost grows linearly in n

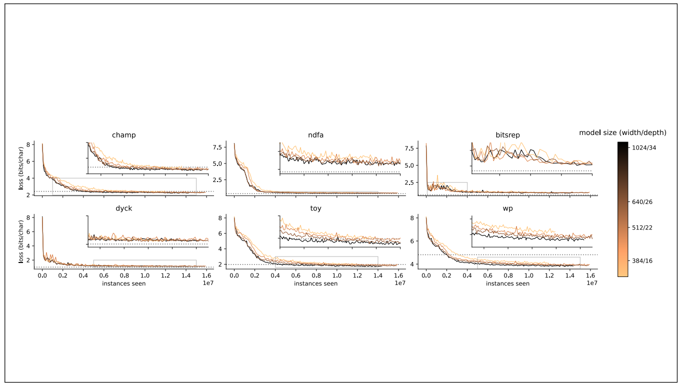
;5;;'6 BUB5CBBBB5Z55BX'X5ZUZZ5P%X555Z5
E$QFGQ.!XQN*Q,.!.,G**GFFFFF^ ^FPQ^!YQF
\^R5D#**JI,DTTTT,TTTS\TITIDSDT*TTTT\\

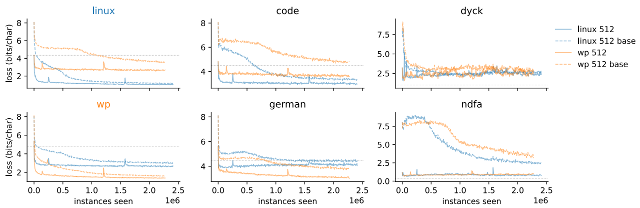


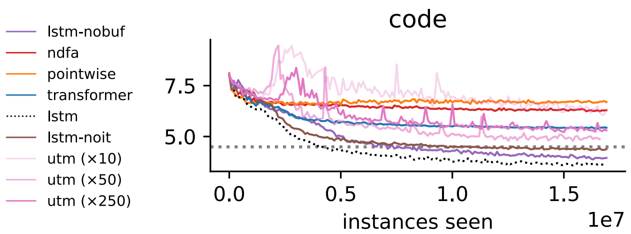
Buffering algorithm


model size (with/depth)
wp
1024/34
640/26
512/22
384/16
loss (bits/char)
instances seen
1e7


random bits flipped
001111111100101010010101
zzààøøzààøzøøøzzzzàà...
x ^ (x + y) + x * y * z + z ^ (x + x)
EXP → EXP + AP, EXP - AP, AP .2, .2, .6
AP → AP * MP, AP / MP, MP .2, .2, .6
MP → GROUP ^ MP, GROUP .2, .8
GROUP → VAR, (EXP). .8, .2
VAR → x, y, z
bitsflip
aut
toy2
german
wp
code
model size (with/depth)
1024/34
640/26
512/22
384/16
loss (bits/char)
instances seen
1e7

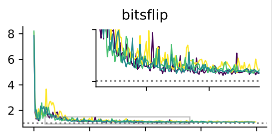

champ
ndfa
bitsrep
dyck
toy
wp
model size (width/depth)
1024/34
640/26
512/22
384/16
loss (bits/char)
instances seen
1e7





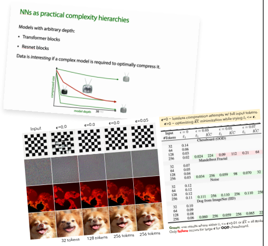
Limitations

- Bias in source model: how universal is it really?
- Poor performance in some simple tasks
- Mix the UTM with the LSTMs?



Future work?

- Identify high-value data (computational depth)
- Generate challenging structured noise *for the current model*.
- Edge-of-chaos models
- Curriculum learning. Build up to high structure.
- Are transformers the right learner?



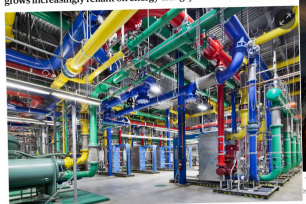
Single-pass Adaptive Image Tokenization for Minimum Program Search, Duggal et al 2025

---

outlook

Google's emissions climb nearly 50% in five years due to AI energy demand

Tech giant's goal of reducing climate footprint at risk as it grows increasingly reliant on energy-hungry data centres



A Google data centre in The Dalles, Oregon, in 2012. Photograph: Google Handout/EPA

Three Mile Island nuclear reactor to restart to power Microsoft AI operations

Pennsylvania plant was site of most serious nuclear meltdown and radiation leak in US history in 1979

The Three Mile Island in 2011. Its owner, Constellation Energy, will restart Unit 1, the reactor it ...sed in 2019. Photograph: Bradley C Bower/AP

nuclear reactor at the notorious Three Mile Island site in Pennsylvania is to

Universal pre-training offers a data / compute tradeoff.