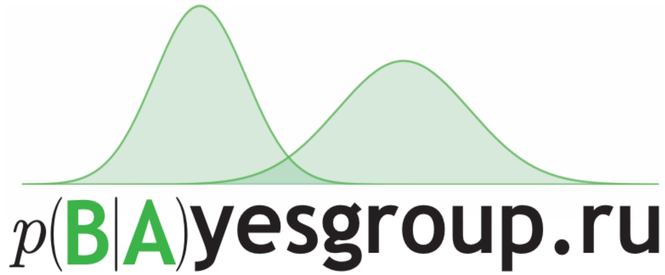
On Power Laws in Deep Ensembles

Ekaterina Lobacheva, Nadezhda Chirkova, Maxim Kodryan,

Dmitry Vetrov

NATIONAL RESEARCH UNIVERSITY

SAMSUNG Research



Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

Global problem setting

When we solve a task with a neural network model we need to choose:

- size of the model - number of parameters
- shape of the model - width / depth / ensembling
- dataset size
- computational budget

We want a recipe!

Our problem setting

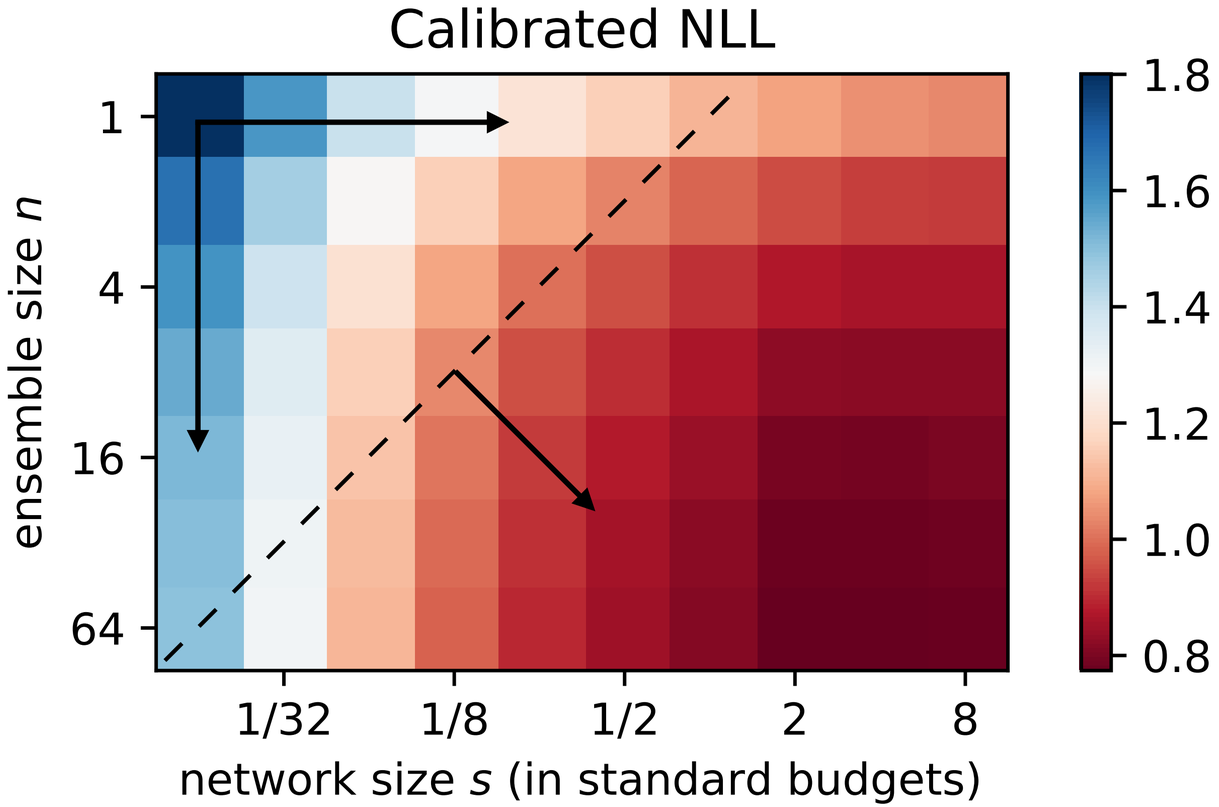
We **do** vary:

- ensemble size
- network size (width)

We **do not** vary:

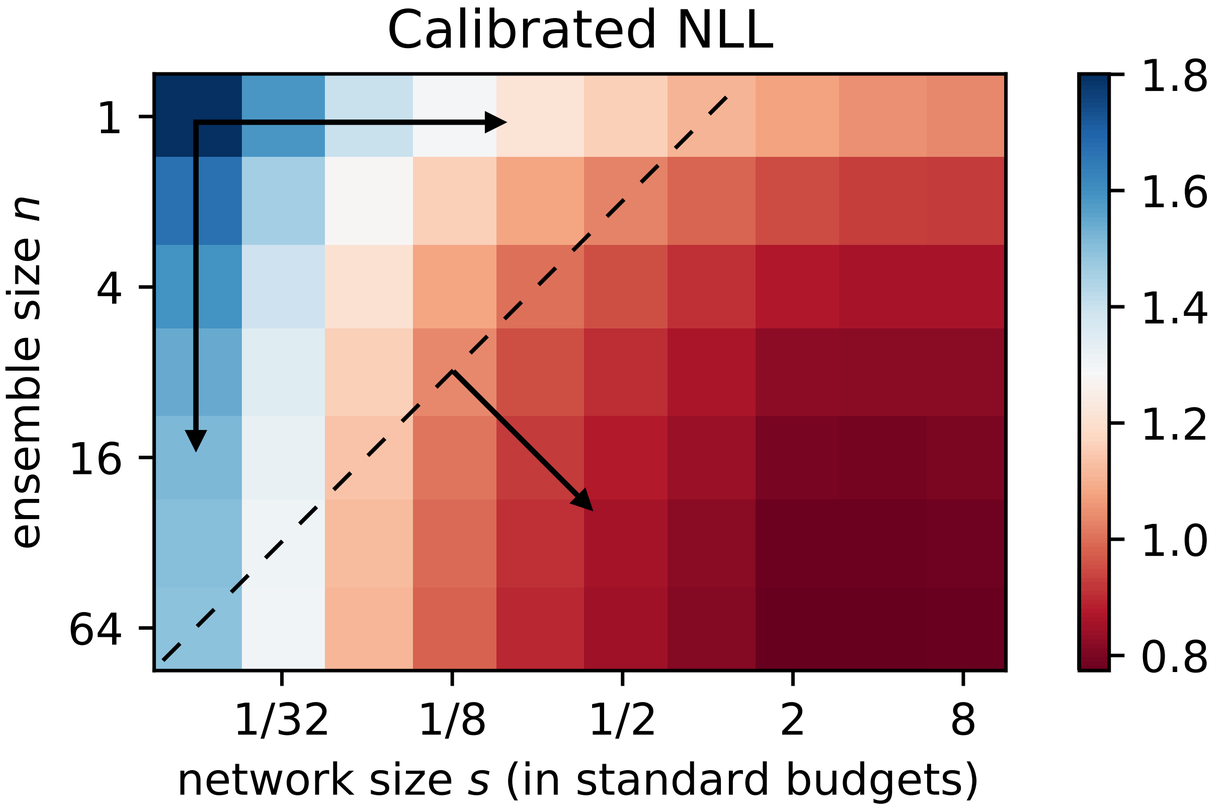
- dataset size
- training procedure

We also fix NLL and CNLL as quality metrics.



Our problem setting: main questions

- How model quality behaves w.r.t. ensemble size, network size and total parameter count?
- Which model to choose if we have a fixed memory budget?
- Is it possible to describe the model quality behaviour with some laws and use these laws to predict quality of large models and optimal ensemble configuration for a fixed memory budget?



Our problem setting: main questions

- How model quality behaves w.r.t. ensemble size, network size and total parameter count? → Power laws
- Which model to choose if we have a fixed memory budget? → MSA effect
- Is it possible to describe the model quality behaviour with some laws and use these laws to predict quality of large models and optimal ensemble configuration for a fixed memory budget? → Yes!

Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

Experimental setup

- Networks: VGG16, WideResNet
- Datasets: CIFAR-10, CIFAR-100
- Wide range of network sizes: from 1/64 to 8 standard budgets (we choose optimal hyper parameters for each model size)
- A lot of networks: for each network size s, we train at least $\ell = \max\{N, 8s_{\text{standard}}/s\}$ networks, N=64/12 for VGG/WideReNet
- All NLL/CNLL values are results of averaging over at least 3 runs

In the presentation all figures are about VGG on CIFAR-100 if not explicitly stated otherwise.

NLL

Model-average NLL of an ensemble of size n:

$$\mathrm{NLL}_n = -\mathbb{E} \sum_{\mathrm{obj} \in \mathcal{D}} \log \bar{p}^*_{\mathrm{obj},n} = -\mathbb{E} \sum_{\mathrm{obj} \in \mathcal{D}} \log \left(\frac{1}{n} \sum_{i=1}^{n} p^*_{\mathrm{obj},i} \right)$$

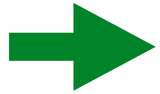
where:

- the expectation in is taken over all possible models
- $p_{\mathrm{obj},i} \in [0,1]^K$ — output probabilities of i-th network from the ensemble
- $\bar{p}_{\mathrm{obj},n} \in [0,1]^K$ — output probabilities of the ensemble
- p^* — probabilities for correct class

CNLL

Ashukha et al. [1]:

Comparison of the NLLs of different models with suboptimal softmax temperature may lead to an arbitrary ranking of the models.



We need calibration!

Model-average CNLL of an ensemble of size n:

$$\mathrm{CNLL}_n = \mathbb{E} \min_{\tau>0} \left\{ - \sum_{\mathrm{obj} \in \mathcal{D}} \log \bar{p}^*_{\mathrm{obj},n}(\tau) \right\}$$

How to apply and choose temperature?

After averaging (classic):

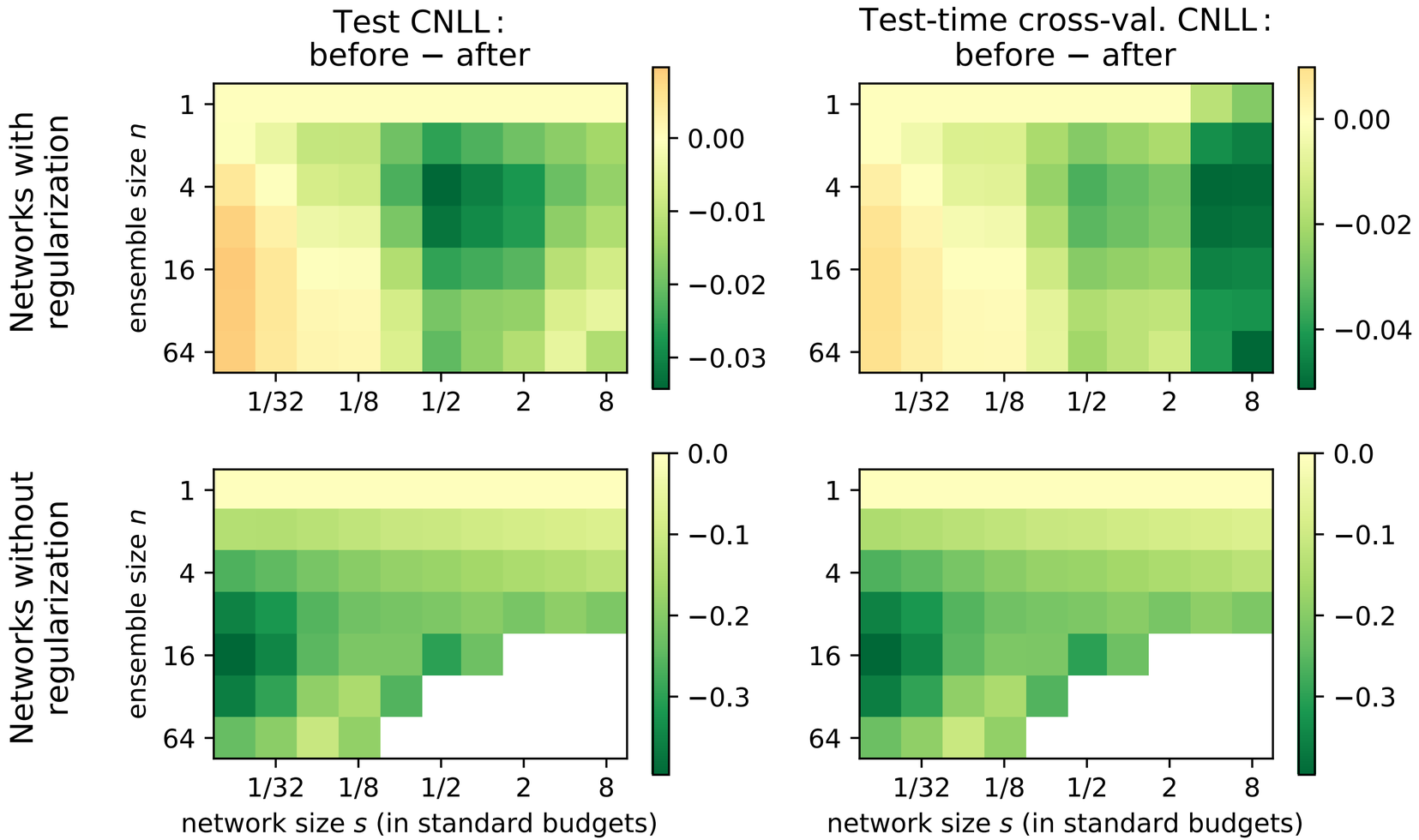
$$\bar{p}_{\mathrm{obj},n}(\tau) = \mathrm{softmax}\Big\{\Big(\log\big(\frac{1}{n}\sum_{i=1}^{n} p_{\mathrm{obj},i}\big)\Big)/\tau\Big\}$$

Before averaging (ours):

$$\bar{p}_{\mathrm{obj},n}(\tau) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{softmax}\{\log(p_{\mathrm{obj},i})/\tau\}$$

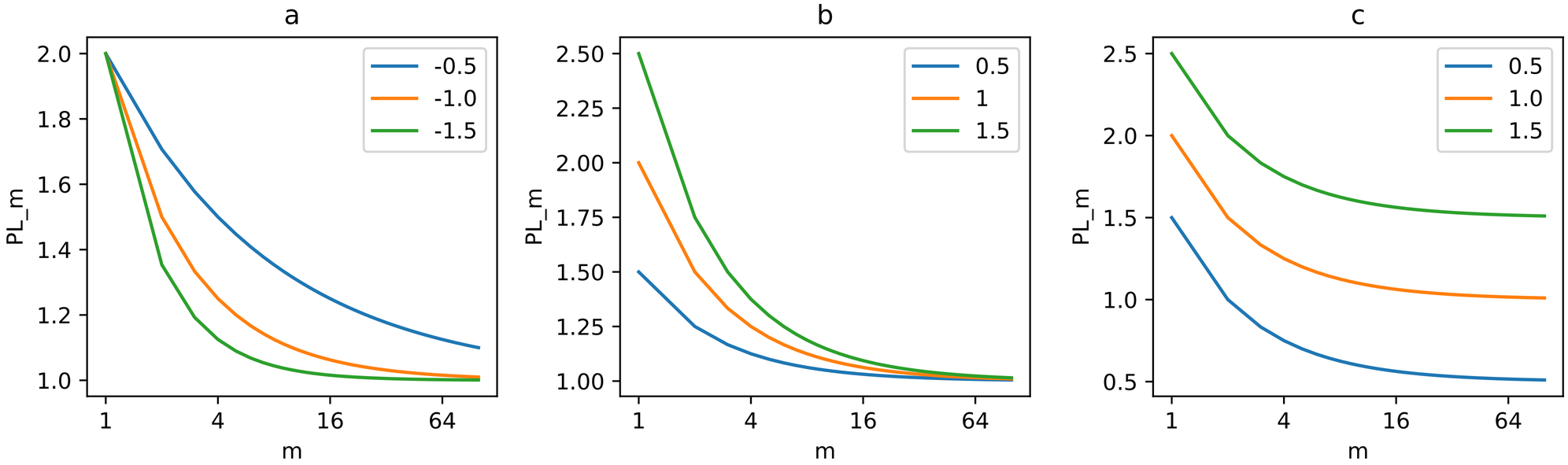
In practice minimization w.r.t. temperature is usually done with test-time cross-validation (we use two equal parts five times).

How to calibrate?



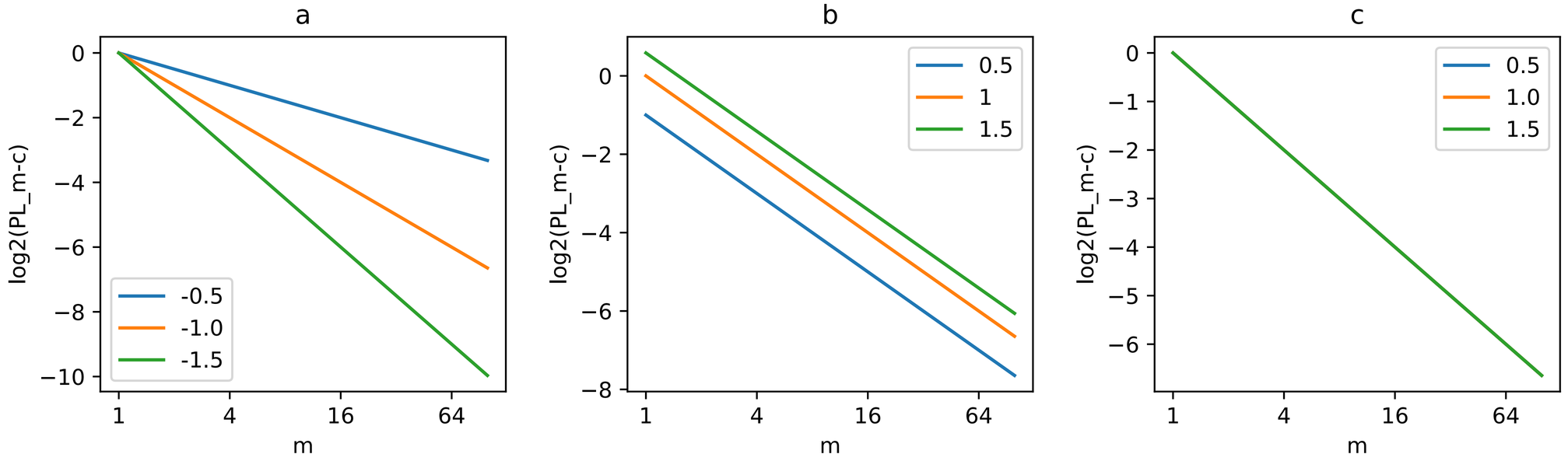
Power law: linear scale

$$\mathrm{PL}_m = c + bm^a$$



Power law: log scale with optimal c

$$\log(\mathrm{PL}_m - c) = \log(bm^a) = a \log m + \log(b)$$



Approximating sequences with power laws

- Given: an arbitrary sequence $\{\hat{y}_m\}$, $m = 1, \ldots, M$
- Approximate with: a power law $\mathrm{PL}_m = c + bm^a$

Optimization problem (we solve it with BFGS):

$$(a, b, c) = \underset{a,b,c}{\operatorname{argmin}} \frac{1}{M} \sum_{m=1}^{M} \left(\log_2(\hat{y}_m - c) - \log_2(bm^a)\right)^2$$

We also reweighs points in some experiments to account for the variable density in logarithmic scale.

Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

Theoretical view: NLL

Proposition 1 *Consider an ensemble of n models, each producing independent and identically distributed probabilities of the correct class for a given object: $p^*_{\text{obj},i} \in [\epsilon_{\text{obj}}, 1]$, $\epsilon_{\text{obj}} > 0$, $i = 1, \ldots, n$. Let $\mu_{\text{obj}} = \mathbb{E}p^*_{\text{obj},i}$ and $\sigma^2_{\text{obj}} = \mathbb{D}p^*_{\text{obj},i}$ be, respectively, the mean and variance of the distribution of probabilities. Then the model-average NLL of the ensemble for a single object can be decomposed as follows:*

$$\text{NLL}_n^{\text{obj}} = \text{NLL}_\infty^{\text{obj}} + \frac{1}{n}\frac{\sigma^2_{\text{obj}}}{2\mu^2_{\text{obj}}} + \mathcal{O}\left(\frac{1}{n^2}\right). \tag{2}$$

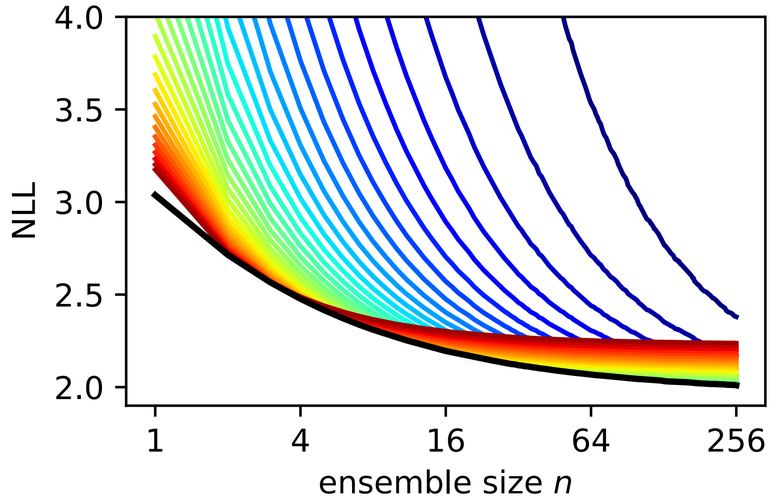
where $\text{NLL}_\infty^{\text{obj}} = -\log\left(\mu_{\text{obj}}\right)$ is the "infinite" ensemble NLL for the given object.

- For $n \to \infty$ NLL_n behaves as power law $c + bn^{-1}$ (similar to [3])
- For finite-range n the dependency may be more complex

Theoretical view: from NLL to CNLL

Steps:

- Apply temperature to NLL and obtain $\mathrm{NLL}_n(\tau)$
- Choose the optimal temperature and obtain the lower envelope $\mathrm{LE\text{-}NLL}_n$
- Understand the difference between practically used CNLL_n and $\mathrm{LE\text{-}NLL}_n$



Theoretical view: NLL with fixed temperature

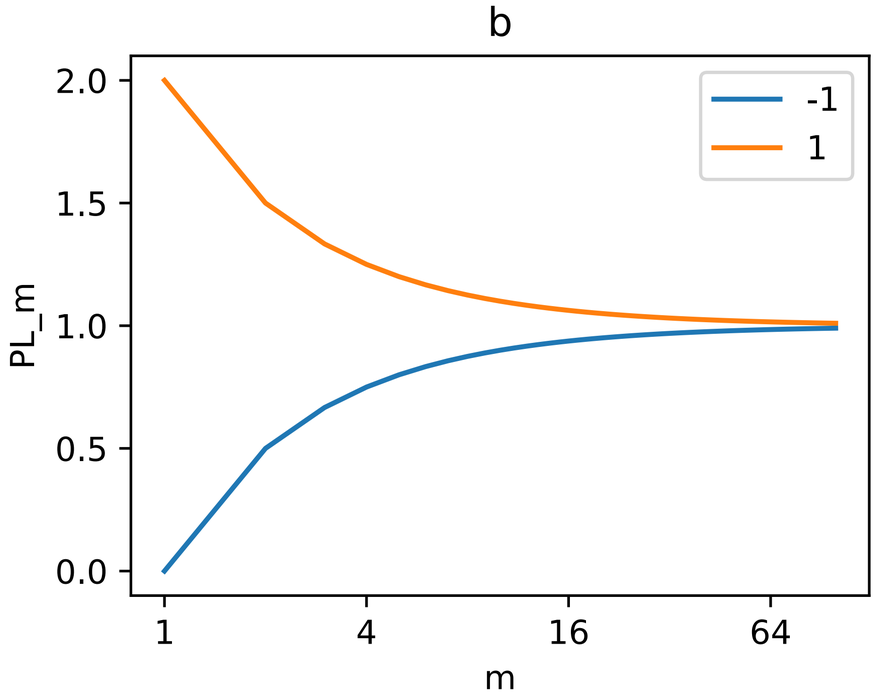
For a fixed temperature $\mathrm{NLL}_n(\tau)$ follows a power law, its form and theoretical grounds are slightly different for two cases:

Temperature before averaging

- Proposition 1 is valid
- Parameter b is always positive

Temperature after averaging

- Proposition needs some adjustment
- Parameter b may be negative



Theoretical view: NLL with fixed temperature

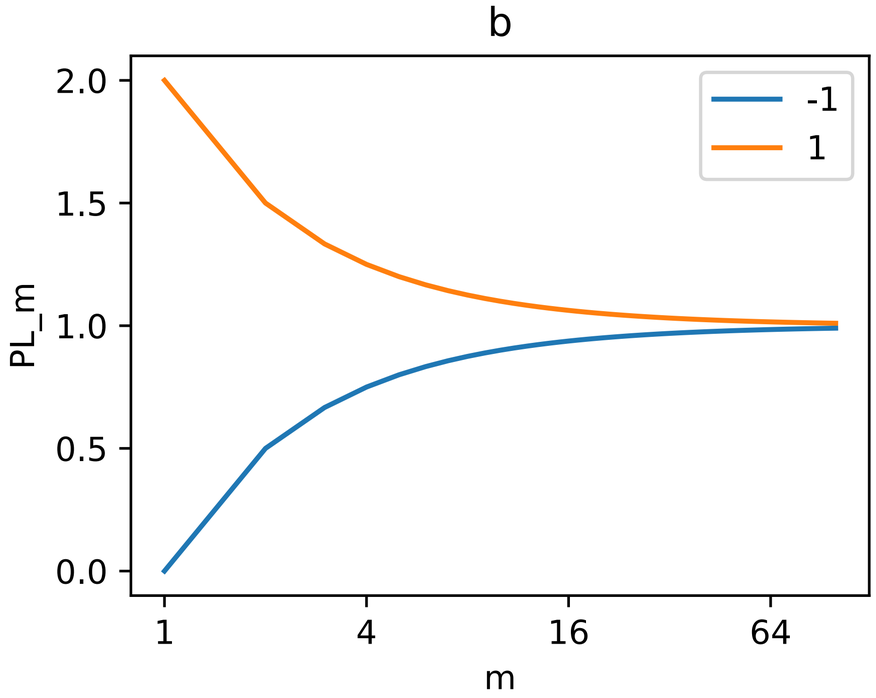
For a fixed temperature $\mathrm{NLL}_n(\tau)$ follows a power law, its form and theoretical grounds are slightly different for two cases:

Temperature before averaging

- Proposition 1 is valid
- Parameter b is always positive

Temperature after averaging

- Proposition needs some adjustment
- Parameter b may be negative

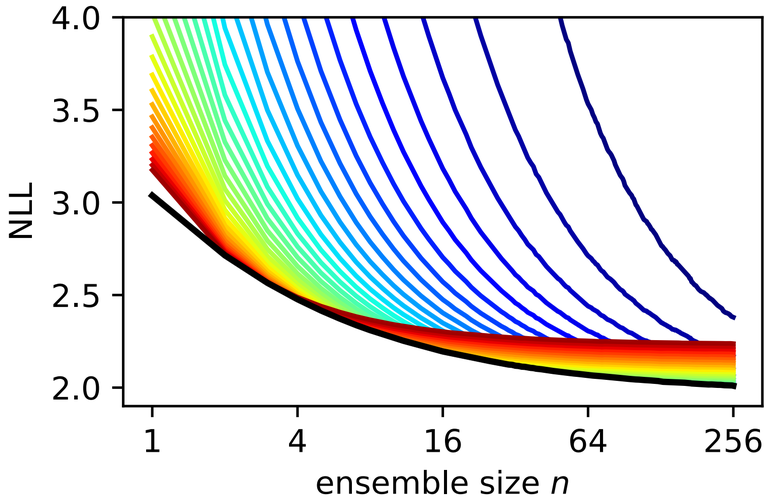


Theoretical view: LE-NLL

Lower envelope of $\mathrm{NLL}_n(\tau)$:

$$\text{LE-NLL}_n = \min_{\tau>0} \mathrm{NLL}_n(\tau), \quad \mathrm{NLL}_n(\tau) \overset{n\to\infty}{\sim} \mathrm{PL}_n$$

- Each $\mathrm{NLL}_n(\tau)$ converges to its asymptote $c(\tau)$
- For large n LE-NLL_n will coincide with $\mathrm{NLL}_n(\tau^*)$



Theoretical view: LE-NLL

Lower envelope of $\mathrm{NLL}_n(\tau)$:

$$\text{LE-NLL}_n = \min_{\tau>0} \mathrm{NLL}_n(\tau), \quad \mathrm{NLL}_n(\tau) \overset{n\to\infty}{\sim} \mathrm{PL}_n$$

- Each $\mathrm{NLL}_n(\tau)$ converges to its asymptote $c(\tau)$
- For large n LE-NLL_n will coincide with $\mathrm{NLL}_n(\tau^*)$

Proposition 2 *Consider a compact set $\mathcal{T}$ and two continuous mappings $c : \mathcal{T} \to \mathbb{R}$ and $b : \mathcal{T} \to \mathbb{R}$. Let each value $\tau \in \mathcal{T}$ correspond to a certain power law w. r. t. n:*

$$\mathrm{PL}_n(\tau) = c(\tau) + \frac{b(\tau)}{n}. \tag{18}$$

Then the lower envelope $\mathrm{LE}_n = \min_{\tau\in\mathcal{T}} \mathrm{PL}_n(\tau)$ of the power laws (18) follows a power law asymptotically.

Theoretical view: LE-NLL vs CNLL

Different order of expectation and minimization:

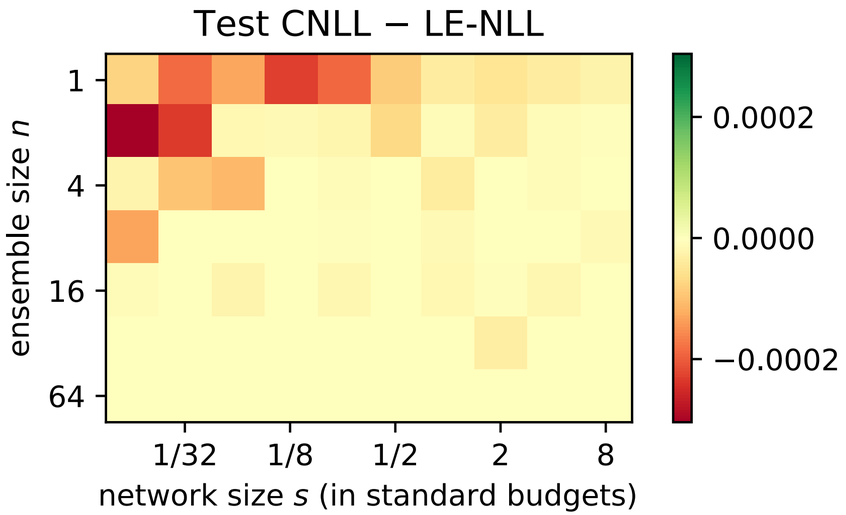
$$\mathrm{LE\text{-}NLL}_n = \min_{\tau>0} \mathbb{E}\left\{ -\sum_{\mathrm{obj}\in\mathcal{D}} \log \bar{p}^*_{\mathrm{obj},n}(\tau) \right\}$$

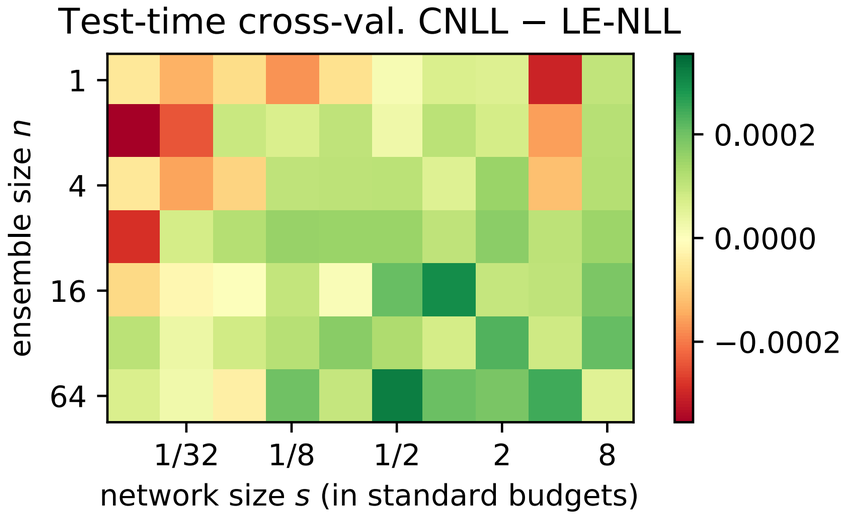
$$\mathrm{CNLL}_n = \mathbb{E} \min_{\tau>0}\left\{ -\sum_{\mathrm{obj}\in\mathcal{D}} \log \bar{p}^*_{\mathrm{obj},n}(\tau) \right\}$$

+ test-time cross-validation is used to compute CNLL in practice.

Theoretical view: LE-NLL vs CNLL

In practice they are very similar:

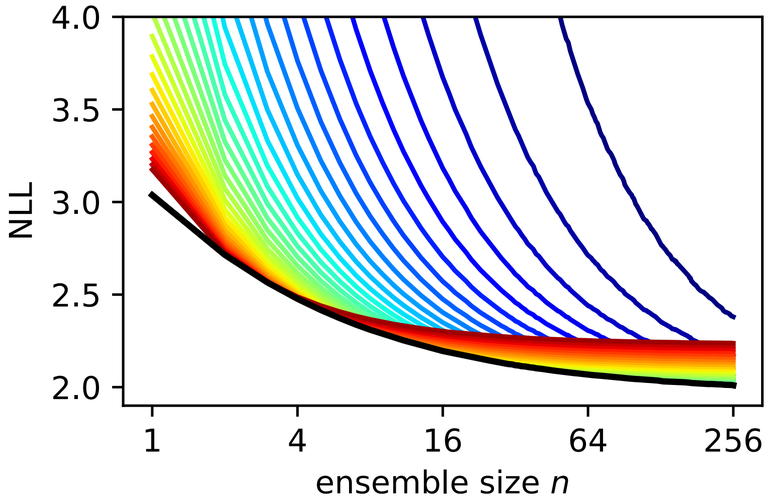




Theoretical view: from NLL to CNLL

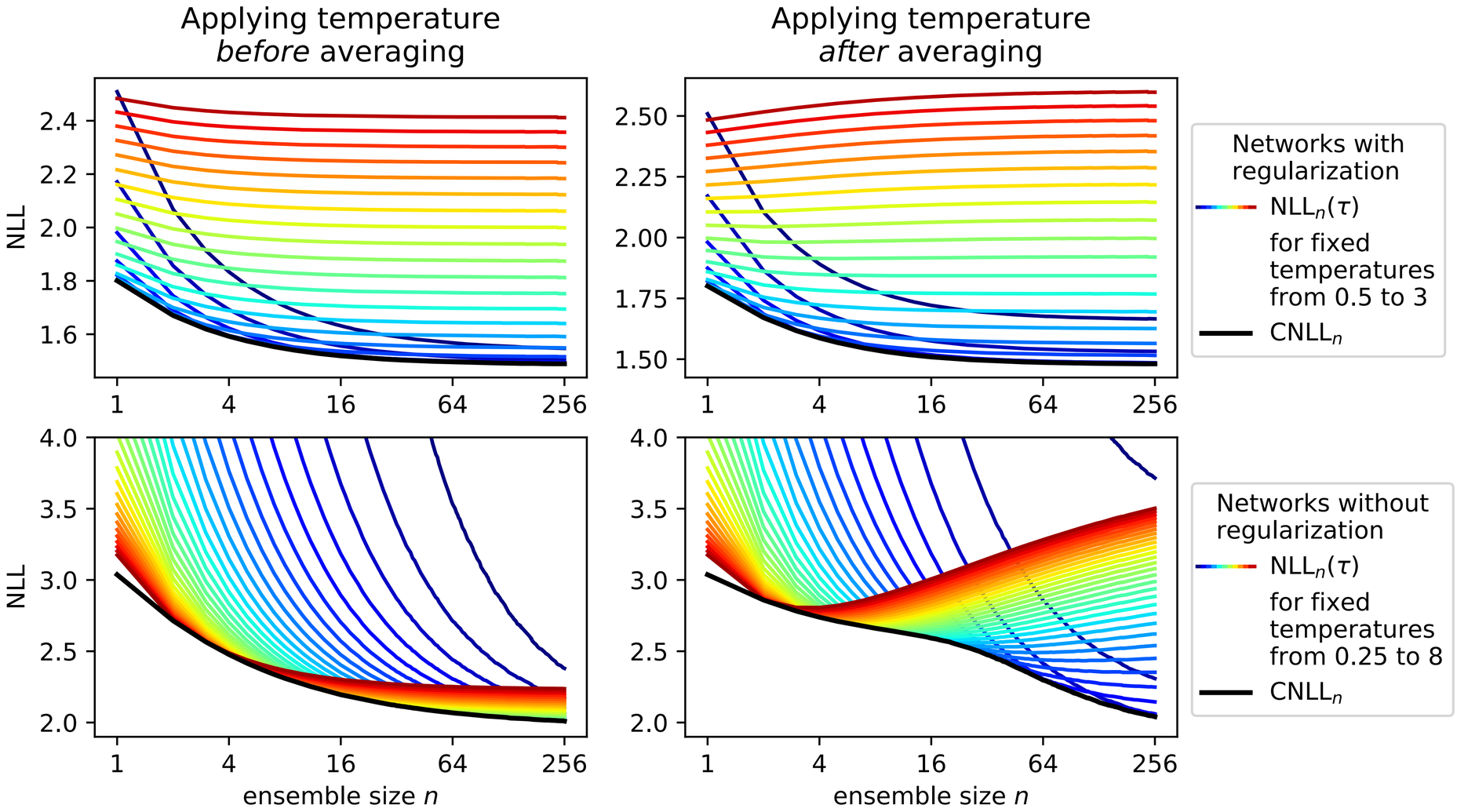
Steps:

- Apply temperature to NLL and obtain $\mathrm{NLL}_n(\tau)$
- Choose the optimal temperature and obtain the lower envelope $\mathrm{LE\text{-}NLL}_n$
- Understand the difference between practically used CNLL_n and $\mathrm{LE\text{-}NLL}_n$



CNLL_n approximately behaves as a power law for $n \to \infty$

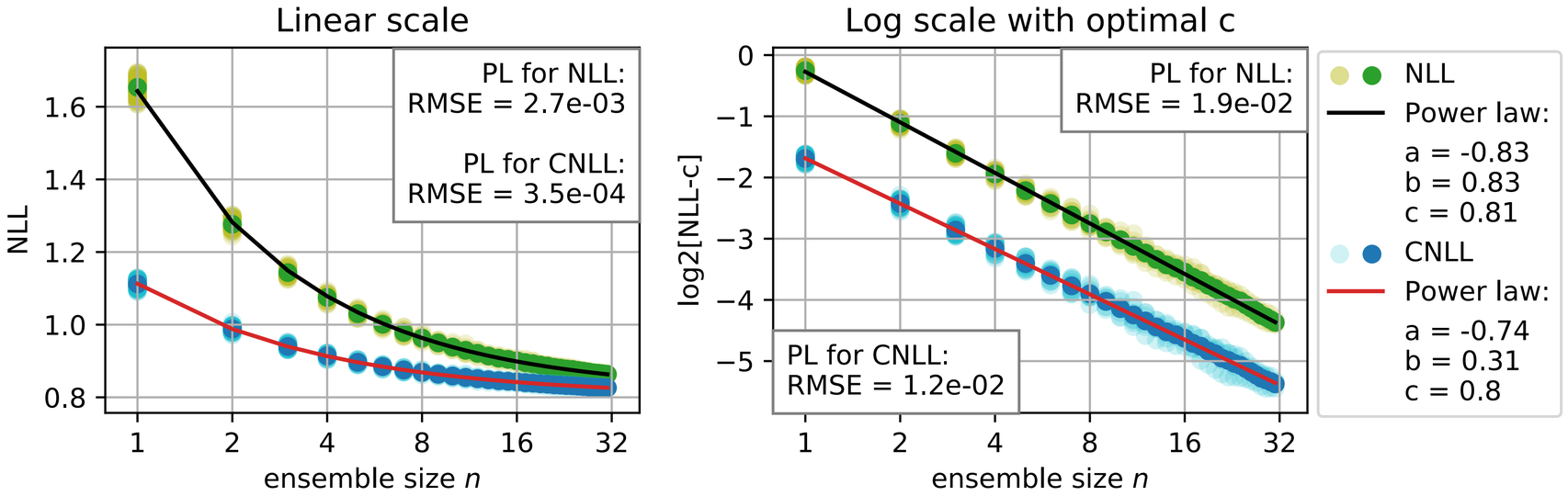
Why we apply temperature before averaging



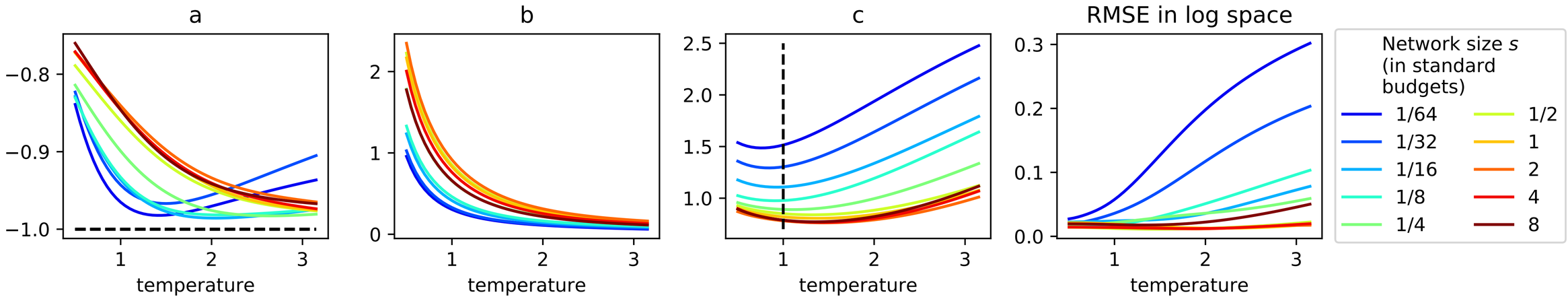
Practical view: NLL and CNLL

In practice we work with finite n, therefore the theoretical results are not applicable and can be used only as inspiration =(

But in practice we also clearly see power laws!

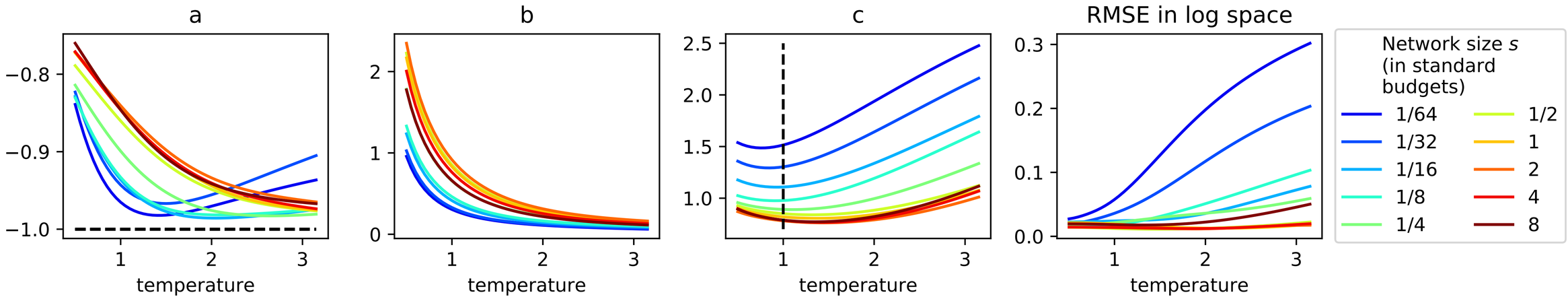


Practical view: NLL with fixed temperature



- Approximation is accurate
- Parameter a is higher than -1 that we saw in theory
- Parameter b reflects the potential gain from the ensembling
- Parameter c approximates the quality of the “infinite”-size ensemble

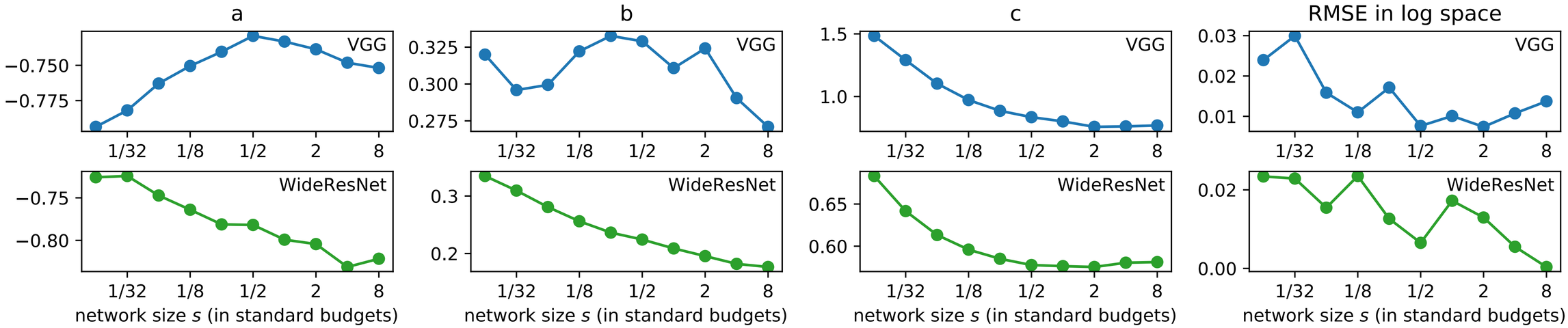
Practical view: NLL with fixed temperature



Interesting observations about c:

- optimal temperature may be lower/higher than 1 => even large ensembles need calibration
- optimal temperature is higher for larger network size => ensembles of large networks are overconfident similarly to large single networks

Practical view: CNLL

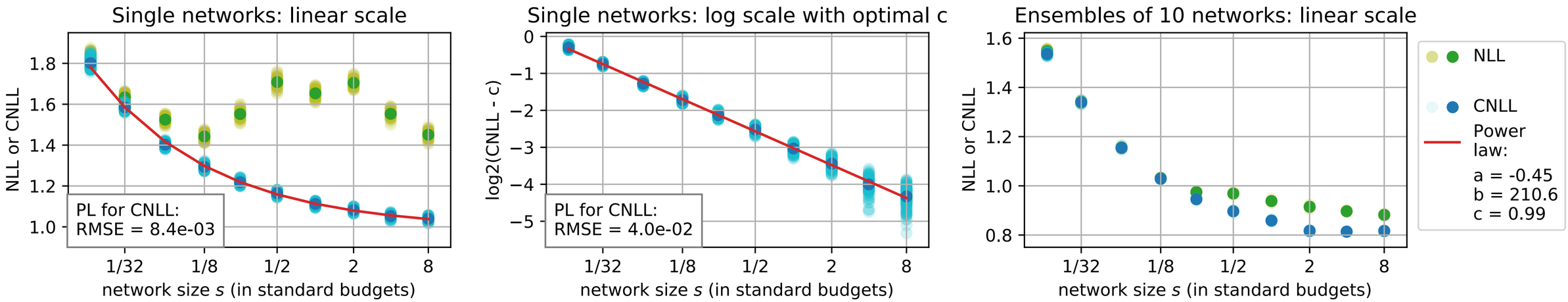


- Approximation is accurate
- Interesting observation: for large s, b decreases and c start to increase. We see two possible reasons:
    - under-regularization
    - decreased diversity of wider networks [2]

Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

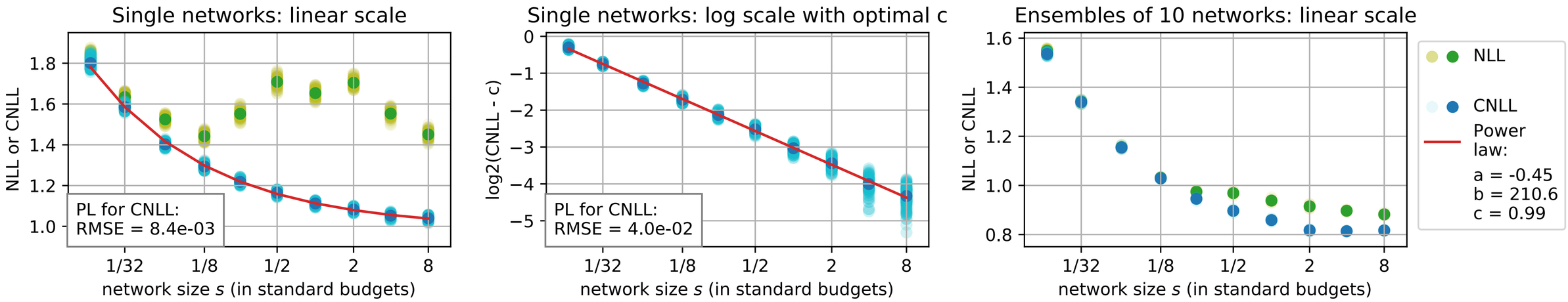
Behaviour w.r.t. network size



Single model:

- Double decent in NLL
- PL for CNLL (interestingly, there is no dd after calibration)
- Parameter a is close to -0.5 as in [3]

Behaviour w.r.t. network size



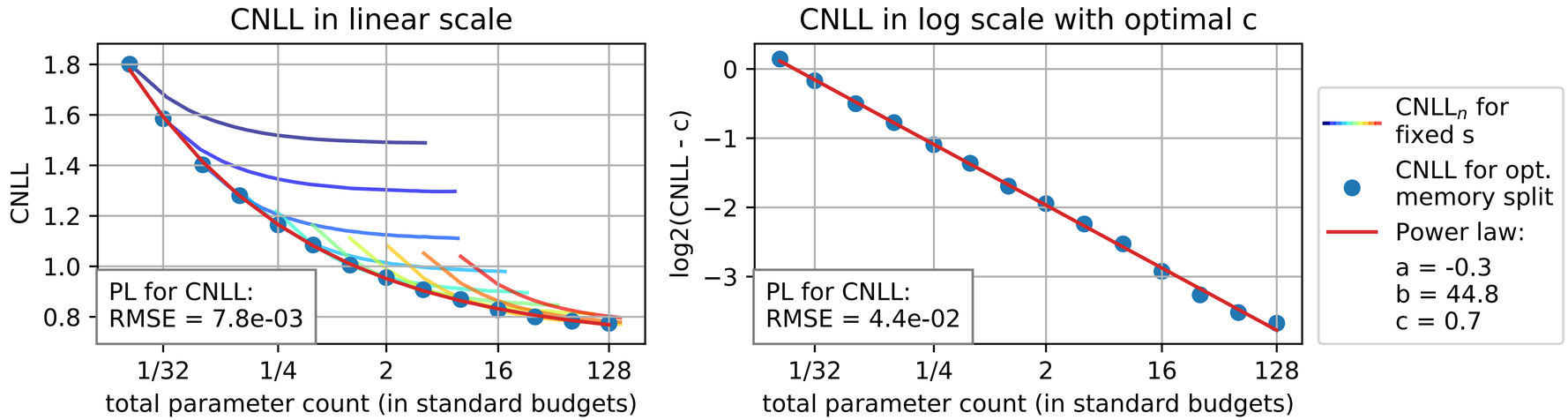
Ensemble:

- Double decent in NLL
- CNLL starts to increase for large s (similarly to the optimal c for CNLL_n)

Outline

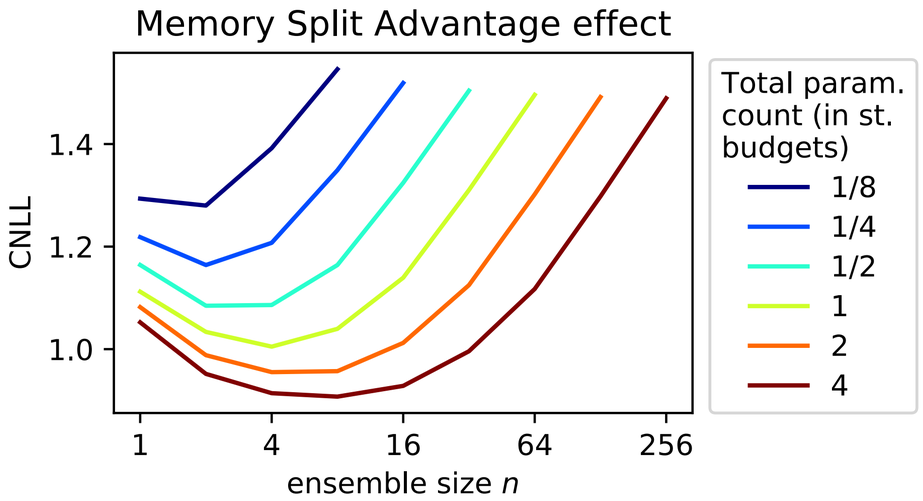
- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

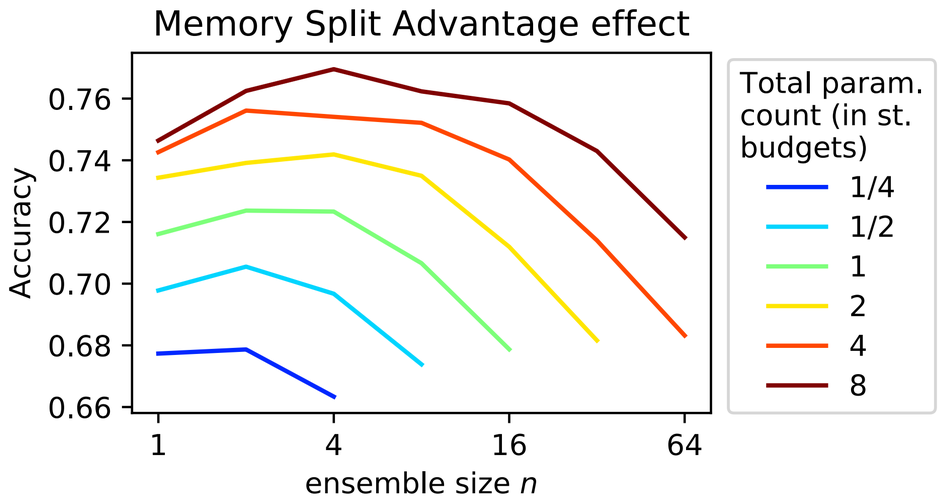
Behaviour w.r.t. total parameter count



- Lower envelope of lower envelopes =)
- PL for CNLL
- Interesting observation: estimation of c becomes more optimistic when we use lower envelopes

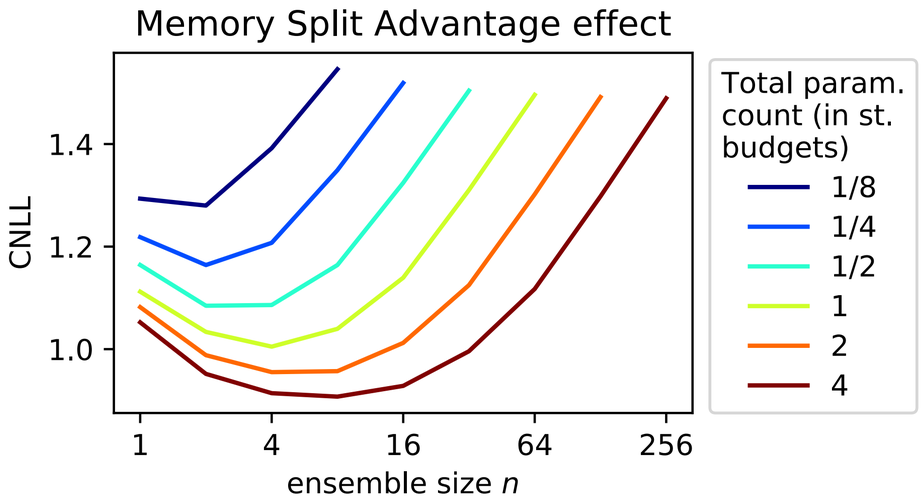
Memory Split Advantage effect

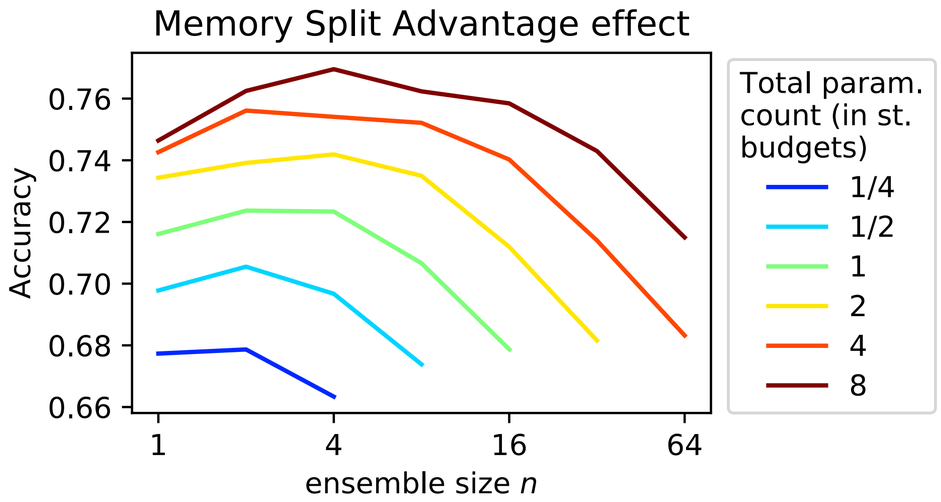




- For each memory budget there is an optimal memory split
- Optimal memory split contains more than 1 network even for budgets smaller than the standard one

Memory Split Advantage effect





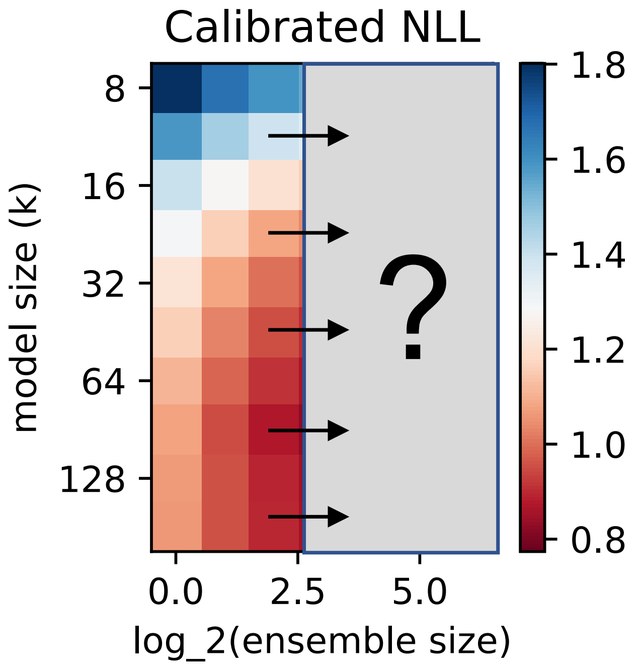
- Memory split is slower in practice but not n times slower, for example: for budget 4S, with a single network / memory split of 4 networks, testing takes 111 / 132 sec, while one training epoch takes 42 / 64 seconds

Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

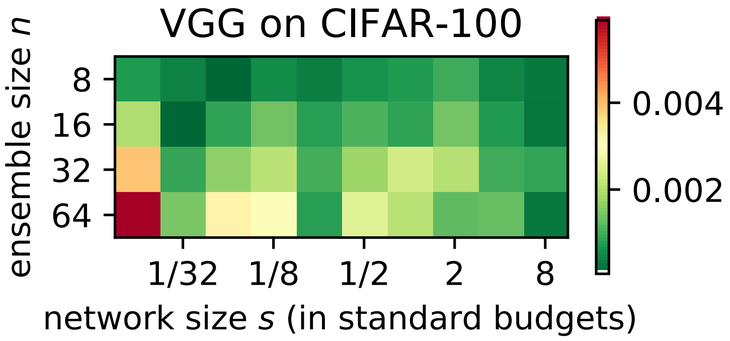
Setup

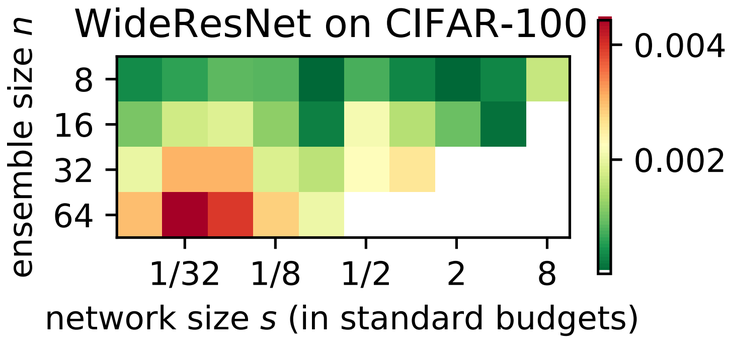
- We take points for ensembles of size 1-4
- How to obtain these points:
  - Ideal-world setting: values are obtained by averaging over a large number of runs (1 experiment)
  - Real-world setting: values are obtained using 6 trained networks of each size (10/5 experiments for VGG/WideResNet)
- What to predict:
  - CNLL for large ensembles
  - Optimal memory splits (using predicted CNLL)



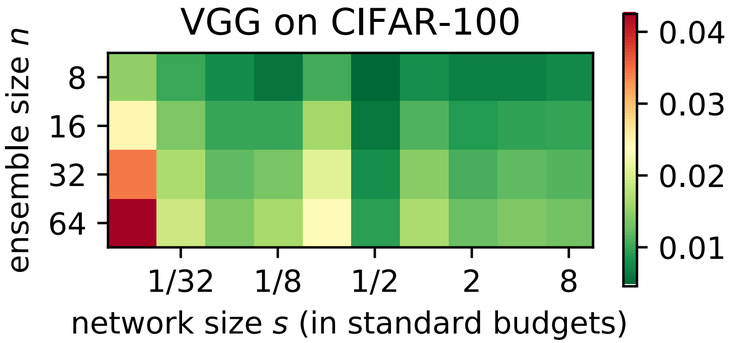
CNLL prediction

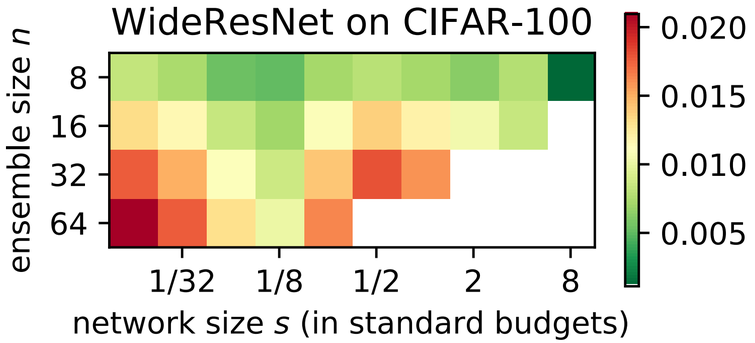
Ideal-world setting: difference between true and predicted CNLL



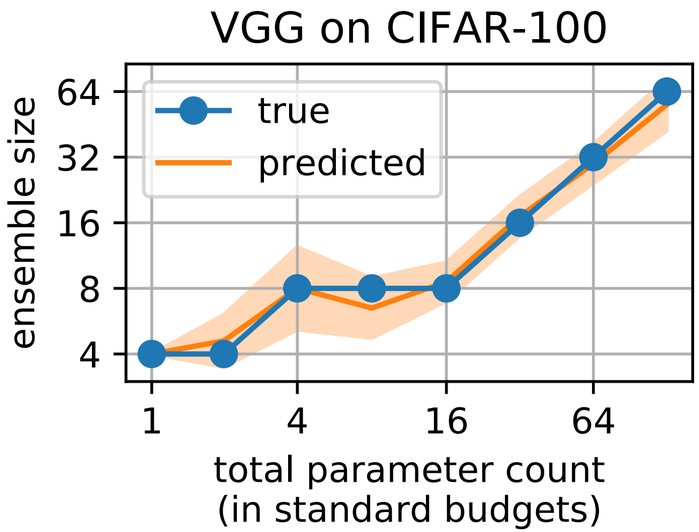


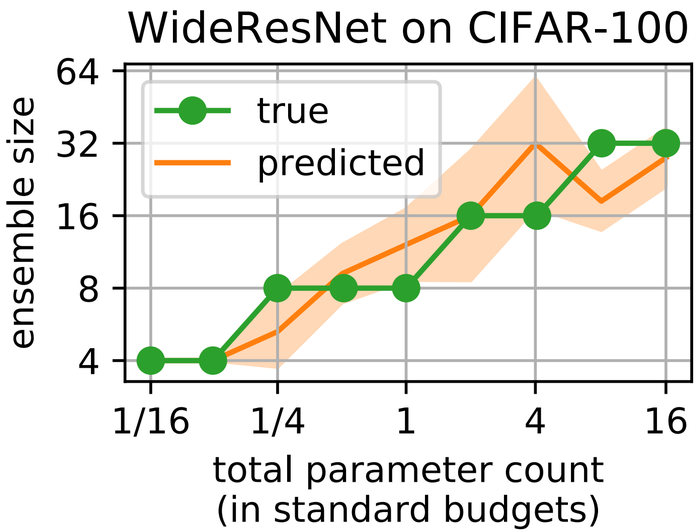
Real-world setting: RMSE between true and predicted CNLL





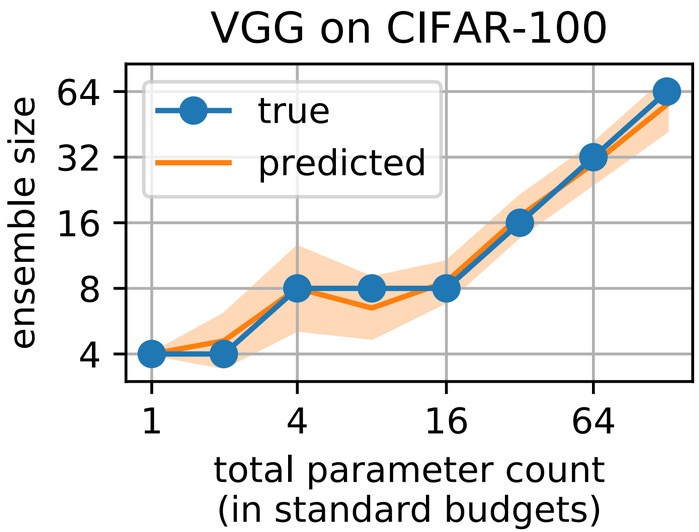
Optimal Memory Split prediction (real-world)

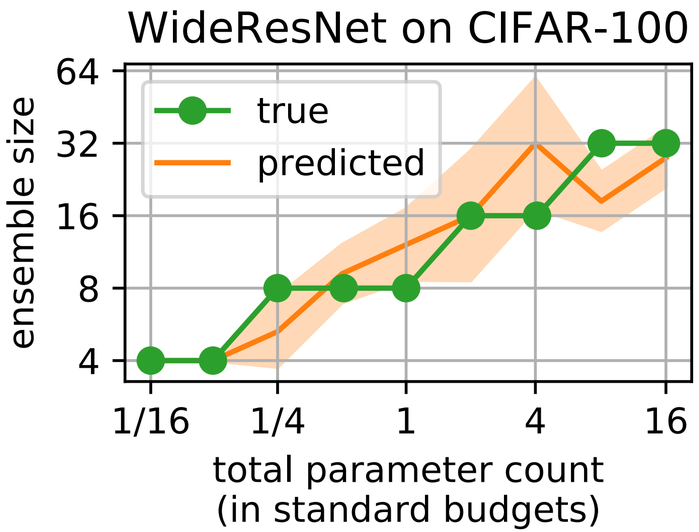




- In most cases we predict either the exact or the neighboring split
- If we predict the neighboring split, the difference in CNLL between the true and predicted splits is negligible (of the same order as the errors presented in previous slide)

Optimal Memory Split prediction (real-world)





- In most cases we predict either the exact or the neighboring split
- If we predict the neighboring split, the difference in CNLL between the true and predicted splits is negligible (of the same order as the errors presented in previous slide)

Conclusion

- NLL:
  - power law w.r.t. ensemble size
  - double decent w.r.t. model size
- CNLL (with temperature applied before averaging):
  - power law w.r.t. ensemble size
  - power law for single model w.r.t. model size and some issues for ensemble of more than 1 network
  - power law w.r.t. total parameter count
- MSA effect for CNLL and accuracy
- Discovered power laws allow prediction of CNLL for large models and optimal memory splits

Outline

- Problem setting
- Notation and experimental setup
- Main part:
  - Behaviour w.r.t. ensemble size
  - Behaviour w.r.t. network size
  - MSA effect and behaviour w.r.t. total parameter count
  - Prediction based on power laws
- What else?

Model and data scaling [4]

- Vary network width and dataset size
- NLL (no calibration), different models/datasets:

(a) Training data size (number of words) and model size (number of parameters excluding word embeddings) for language modeling tasks.

Dataset	Size (N)	Scales (n)	Base Model	Size (M)	Scales (m)
PTB	0.9M	$2^{-k}N$, $0 \leq k \leq 5$	AWD-LSTM	20M	$4^{-k}M$, $0 \leq k \leq 6$
WikiText-2	2M		AWD-LSTM	20M	
WikiText-103	100M		Transformer-XL	41M	

(b) Training data size (number of images) and model size (number of parameters) for image classification tasks.

Dataset	Size (N)	Scales (n)	Base Model	Size (M)	Scales (m)
ImageNet	1.2M	$2^{-k}N$, $0 \leq k \leq 6$	ResNet-50	25.5M	$4^{-k}M$, $0 \leq k \leq 6$
CIFAR10	60K	$2^{-k}N$, $0 \leq k \leq 5$	WRN-44-16	0.7M	$4^{-k}M$, $-3 \leq k \leq 4$
CIFAR100	60K		WRN-44-16	0.7M	$4^{-k}M$, $-2 \leq k \leq 4$
DTD	5640		WRN-44-16	0.7M	
Aircraft	10K		WRN-44-16	0.7M	
UCF101	13K		WRN-44-16	0.7M	

Model and data scaling [4]

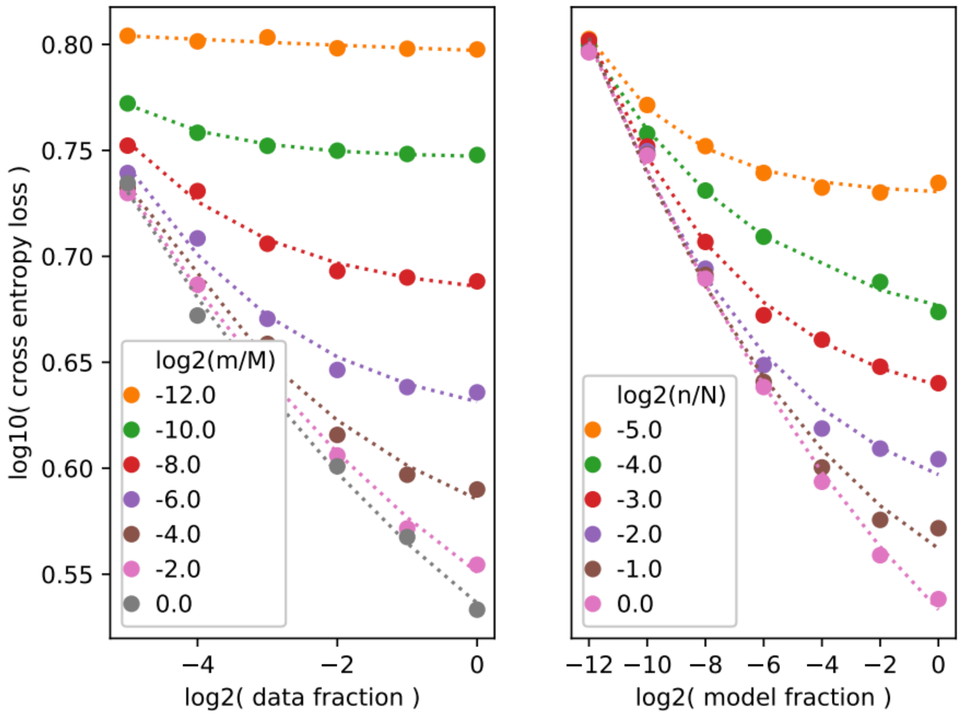
- Empirically find power law like dependencies (m - model size, n- data size):

$$\epsilon(m,n) \approx b(n)m^{-\beta(n)} + c_m(n)$$

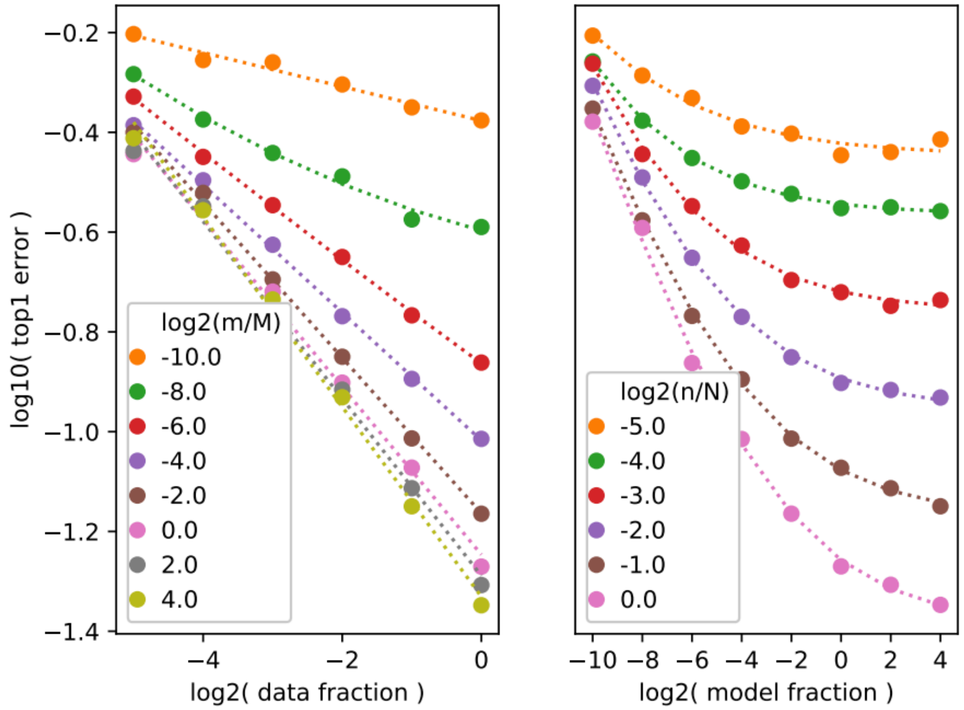
$$\epsilon(m,n) \approx a(m)n^{-\alpha(m)} + c_n(m)$$

Model and data scaling [4]

- Empirically find power law like dependencies (m - model size, n- data size):



(a) Wiki103 cross entropy vs. data and model size.



(b) CIFAR10 top1 error vs. data and model size.

Model and data scaling [4]

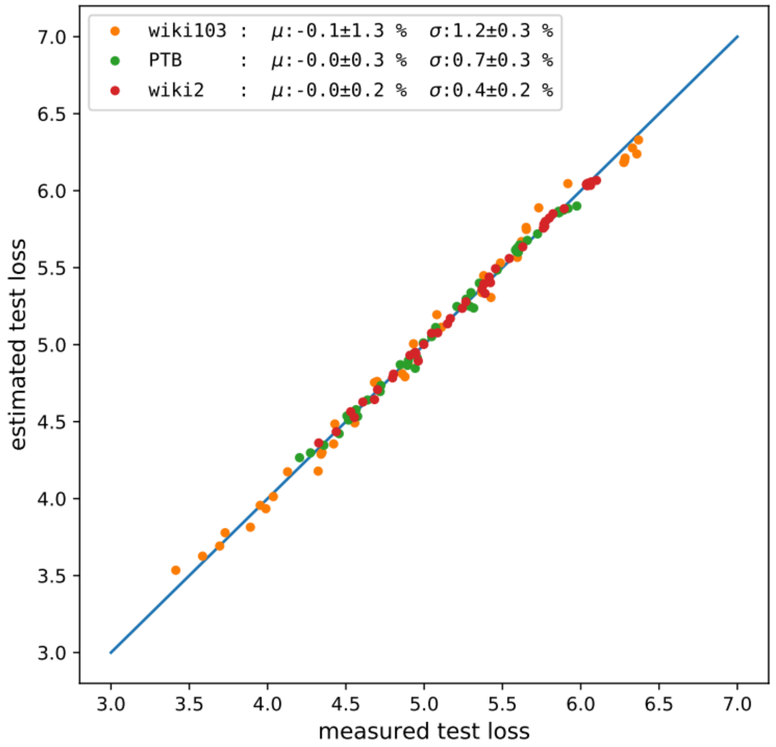
- Empirically find power law like dependencies (m - model size, n- data size):

$$\epsilon(m,n) \approx a(m)n^{-\alpha(m)} + b(n)m^{-\beta(n)} + c_\infty$$

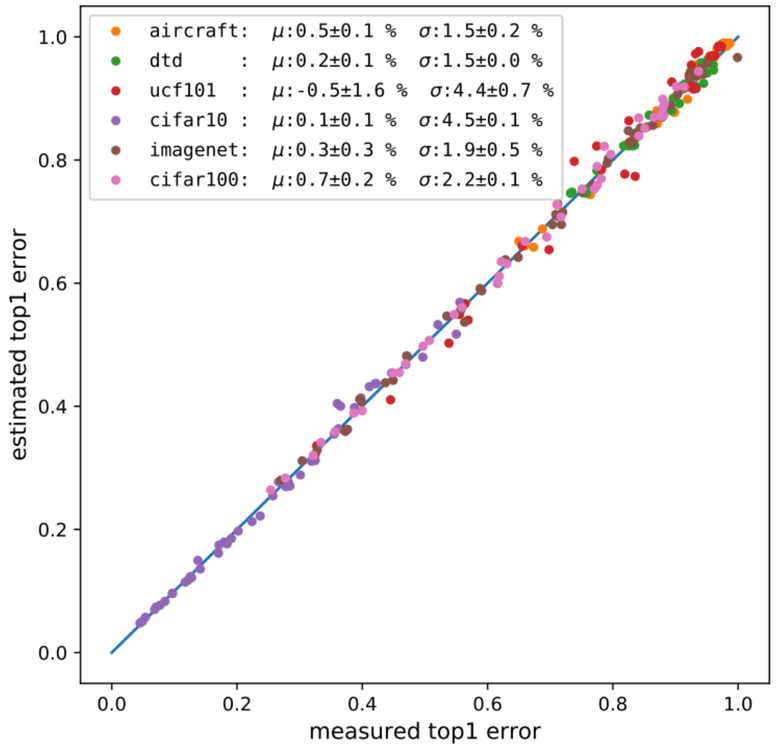
$$\hat{\epsilon}(m,n) = \epsilon_0 \left\| \frac{\tilde{\epsilon}(m,n)}{\tilde{\epsilon}(m,n) - i\eta} \right\| = \epsilon_0 \left\| \frac{an^{-\alpha} + bm^{-\beta} + c_\infty}{an^{-\alpha} + bm^{-\beta} + c_\infty - i\eta} \right\|$$

Model and data scaling [4]

- Empirically find power law like dependencies (m - model size, n- data size):



(a) Estimated vs. actual cross-entropy loss for various language modeling datasets.



(b) Estimated vs. actual test error for various image classification datasets.

Model and data scaling [4]

- Empirically find power law like dependencies

+ some experiments for depth-scaling
+ extrapolation experiments
+ design advise on optimal ratio of model to data size

Scaling Laws for NLP [5]

- Transformers on WebText2 (language modelling)
- NLL (no calibration)
- Vary network size and shape, dataset size, batch size, training compute budget; empirically find power laws:

Parameters	Data	Compute	Batch Size	Equation
N	∞	∞	Fixed	$L(N) = (N_c/N)^{\alpha_N}$
∞	D	Early Stop	Fixed	$L(D) = (D_c/D)^{\alpha_D}$
Optimal	∞	C	Fixed	$L(C) = (C_c/C)^{\alpha_C}$ (naive)
N_{opt}	D_{opt}	C_{min}	$B \ll B_{\text{crit}}$	$L(C_{\text{min}}) = \left(C_c^{\text{min}}/C_{\text{min}}\right)^{\alpha_C^{\text{min}}}$
N	D	Early Stop	Fixed	$L(N, D) = \left[\left(\frac{N_c}{N}\right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D}\right]^{\alpha_D}$
N	∞	S steps	B	$L(N, S) = \left(\frac{N_c}{N}\right)^{\alpha_N} + \left(\frac{S_c}{S_{\text{min}}(S,B)}\right)^{\alpha_S}$

Scaling Laws for NLP [5]

Practical conclusions:

- Performance depends strongly on scale, weakly on model shape

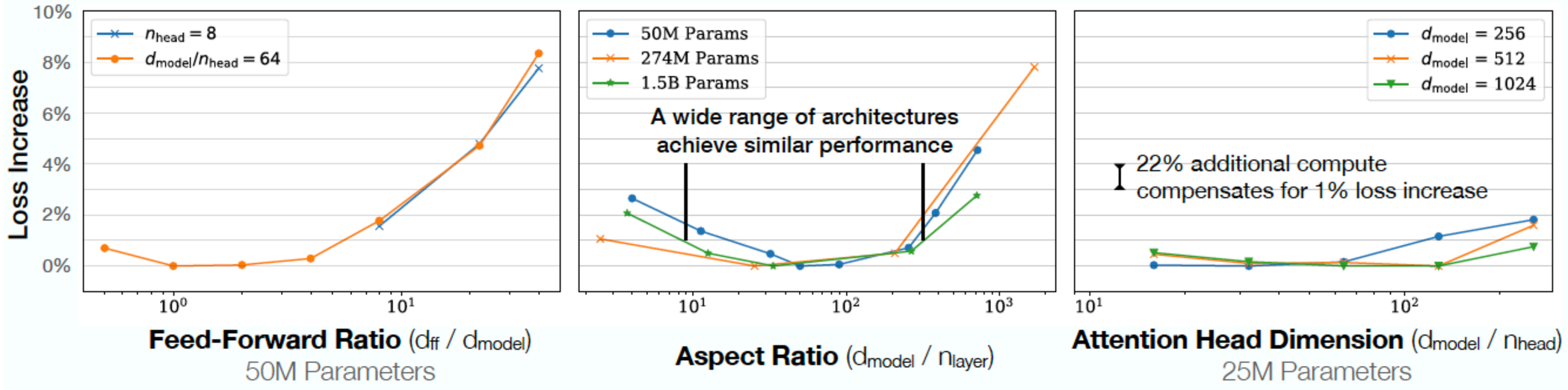


Figure 5 Performance depends very mildly on model shape when the total number of non-embedding parameters N is held fixed. The loss varies only a few percent over a wide range of shapes. Small differences in parameter counts are compensated for by using the fit to $L(N)$ as a baseline. Aspect ratio in particular can vary by a factor of 40 while only slightly impacting performance; an $(n_{\text{layer}}, d_{\text{model}}) = (6, 4288)$ reaches a loss within 3% of the $(48, 1600)$ model used in [RWC+19].

Scaling Laws for NLP [5]

Practical conclusions:

- Power laws for N/C/D (when not bottlenecked by the other two):

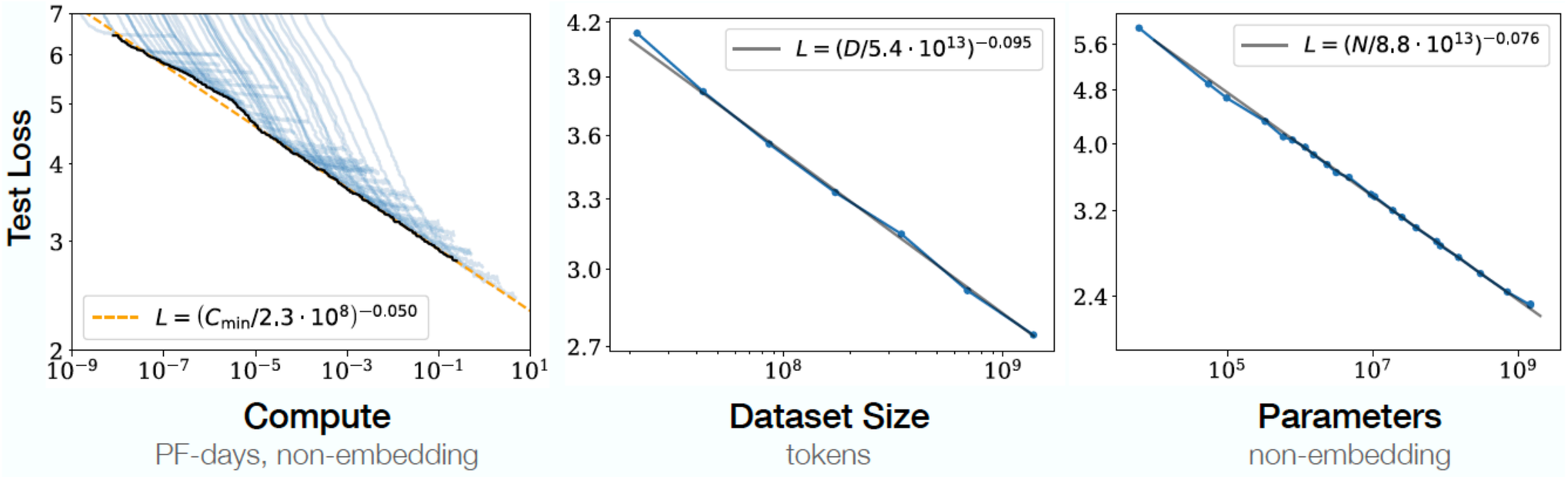
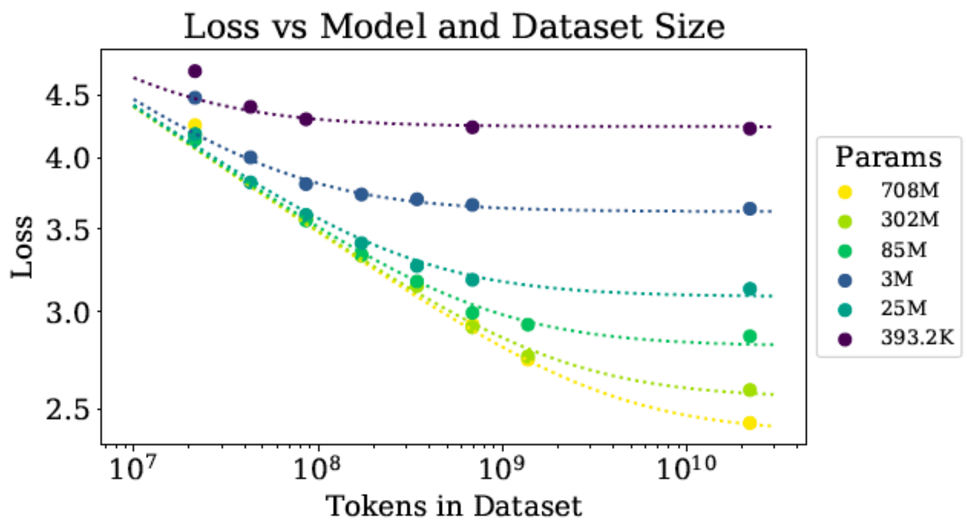


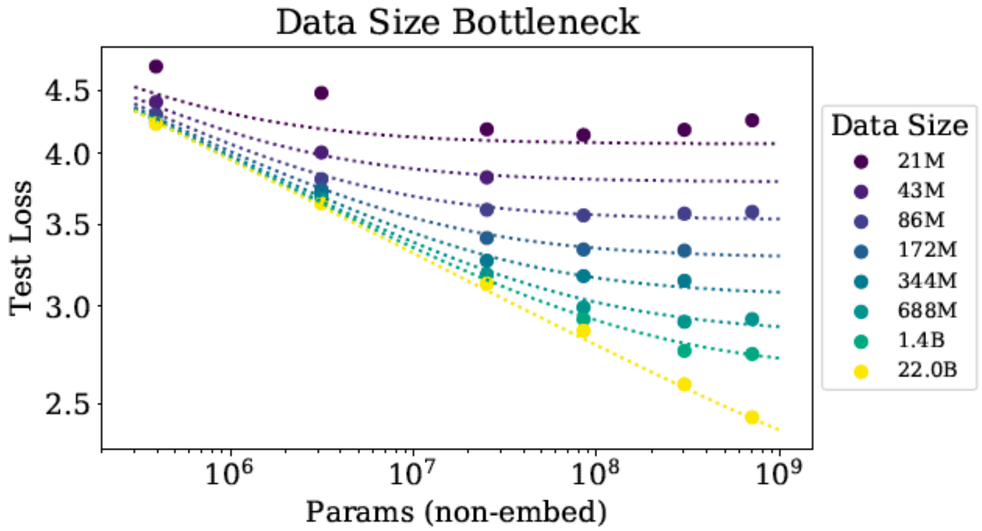
Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

Scaling Laws for NLP [5]

Practical conclusions:

- Power laws for N/C/D (when not bottlenecked by the other two) — but they do not account for our parameter c + global parameters give not very accurate fit in some regions

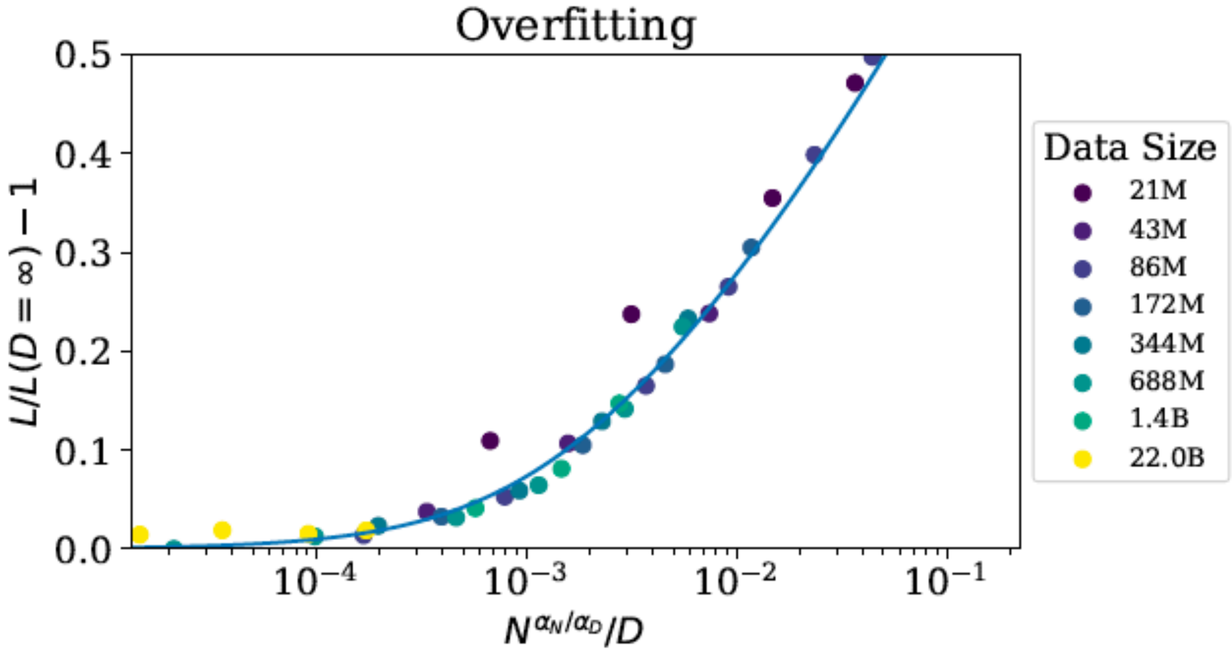




Scaling Laws for NLP [5]

Practical conclusions:

- Universality of overfitting: the performance penalty depends predictably on the ratio $N^{0.74}/D$
- Optimal ratio between model and data size: $D \gtrsim (5 \times 10^3)\, N^{0.74}$



Scaling Laws for NLP [5]

Practical conclusions:

- Universality of training: training curves follow predictable power-laws whose parameters are roughly independent of the model size

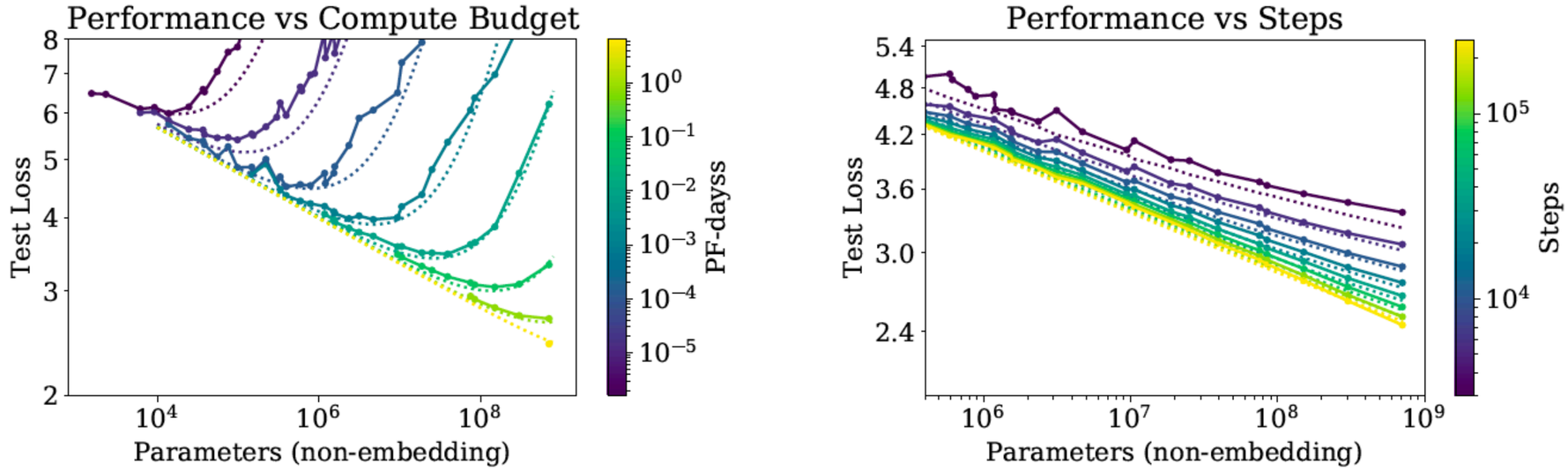
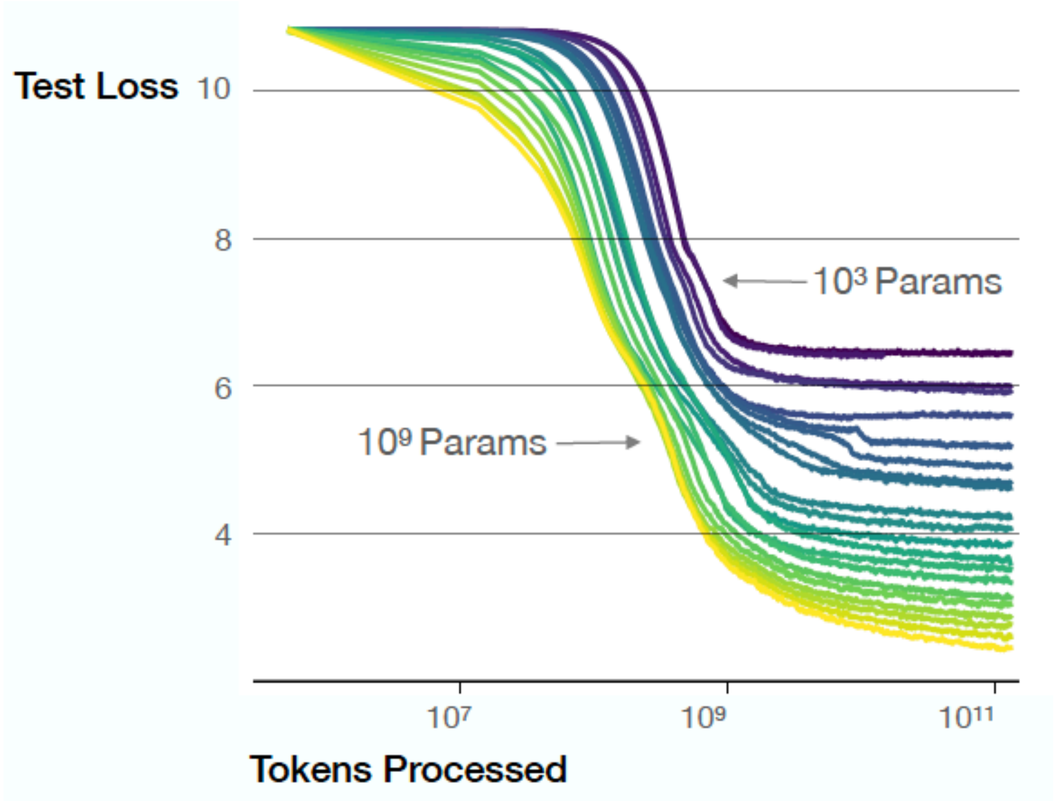


Figure 11 When we hold either total compute or number of training steps fixed, performance follows $L(N, S)$ from Equation (5.6). Each value of compute budget has an associated optimal model size that maximizes performance. Mediocre fits at small S are unsurprising, as the power-law equation for the learning curves breaks down very early in training.

Scaling Laws for NLP [5]

Practical conclusions:

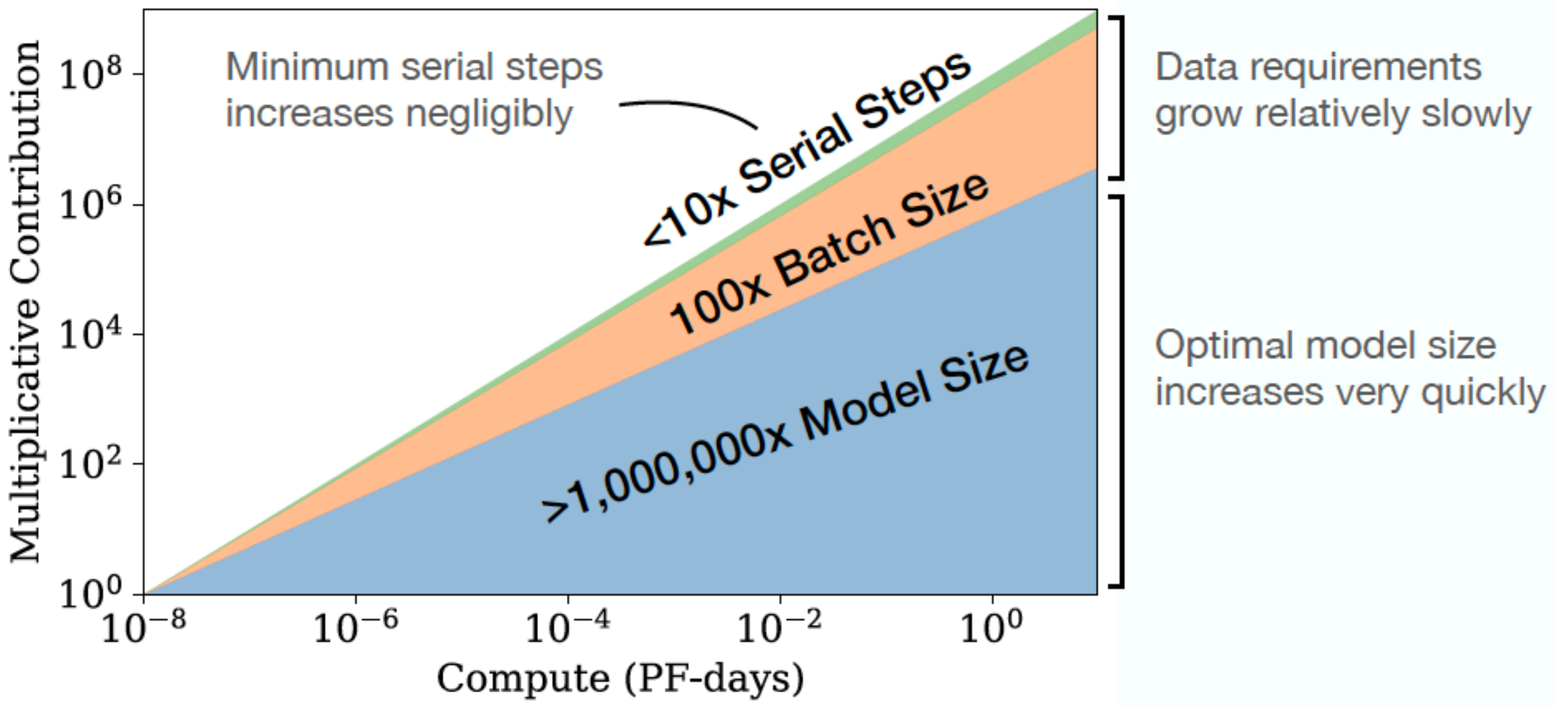
- Sample efficiency: large models are more sample-efficient than small models, reaching the same level of performance with fewer optimization steps and using fewer data points



Scaling Laws for NLP [5]

Practical conclusions:

- How to spend compute budget (example for billion-fold increase):



How to build NN ensembles [6]

- Networks: VGG, ResNet, DenseNet, and WideResNets
- Datasets: SVHN, CIFAR-10, CIFAR-100, and Tiny ImageNet
- Vary width/depth/number of networks
- Reopen MSA effect and claim that they are the first ones =(

References

[1] Ashukha, A., Lyzhov, A., Molchanov, D., and Vetrov, D. (2020). Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. ICLR.
[2] Neal, B., Mittal, S., Baratin, A., Tantia, V., Scicluna, M., Lacoste-Julien, S., and Mitliagkas, I. (2018). A modern take on the bias-variance tradeoff in neural networks. arXiv preprint arXiv:1810.08591.
[3] Geiger, M., Jacot, A., Spigler, S., Gabriel, F., Sagun, L., d'Ascoli, S., Biroli, G., Hongler, C., and Wyart, M. (2020). Scaling description of generalization with number of parameters in deep learning. Journal of Statistical Mechanics: Theory and Experiment, 2020(2):023401.
[4] Jonathan S. Rosenfeld and Amir Rosenfeld and Yonatan Belinkov and Nir Shavit (2020). A Constructive Prediction of the Generalization Error Across Scales. ICLR.
[5] Jared Kaplan and Sam McCandlish and Tom Henighan and Tom B. Brown and Benjamin Chess and Rewon Child and Scott Gray and Alec Radford and Jeffrey Wu and Dario Amodei (2020). Scaling Laws for Neural Language Models. https://arxiv.org/abs/2001.08361
[6] ICLR 2021 submit - https://openreview.net/forum?id=z5Z023VBmDZ