Singular Learning Theory 20 - In-context learning

The nature of in-context learning in Transformer models is likely to deeply inform any future theory which links the geometry of singularities in Transformers as learning machines to the emergent phenomena such as reasoning and abstraction which make these systems interesting.

In each layer of a Transformer model [PH] a list of entity representations $e_1, \ldots, e_\ell \in V$ are processed to form new representations $e_1', \ldots, e_\ell'$. This processing depends on the weights of the Transformer, in several ways

(i) The layer normalisation
(ii) The attention mechanism
(iii) The feedforward layer in each block.

The final prediction also depends on embedding and unembedding weights, which we ignore here. We focus on (ii) the <u>attention weights</u> and how they are changed by backpropagation. The claim in [ICL1, ICL2] and other papers is that <u>in-context learning is "similar" to this change in attention weights</u>. These papers are not to be taken seriously, but let us investigate the question.

<u>The cartoon</u> Let T_W denote the Transformer model with attention weights W, as a family of functions $\{T_W^{(n)}\}_{n \geq 1}$ where $T_W^{(n)}$ takes a sequence of tokens $\underline{x} = x_1, \ldots, x_n$ as input and returns the prediction of the next token as a probability distribution. So the contribution of a sequence $x_1, \ldots, x_\ell, x_{\ell+1}$ to a gradient step is

$$\begin{aligned} \Delta W &= -\eta \nabla_W \left(-\log T_W^{(n)}(\underline{x})[x_{\ell+1}] \right) \\ &\propto \nabla_W T_W^{(n)}(\underline{x})[x_{\ell+1}] \end{aligned} \qquad (1.1)$$

Suppose the tokens were <u>examples</u> of a task

$$\underbrace{x_1, \dots, x_l}_{\text{question}} \qquad \underbrace{x_{l+1}}_{\text{answer}} \tag{2.1}$$

Then in-context learning refers to the possibility that on a second question $y_1, \dots, y_l$

$$\underset{\text{"learned from } \underline{x}\text{"}}{T_W^{(2l+1)}(\underline{x}, x_{l+1}, \underline{y})} \text{ is closer to } y_{l+1} \text{ than } T_W^{(l)}(\underline{y}) \tag{2.2}$$

There is of course another way to "improve" T_W given $\underline{x}, x_{l+1}$ which is to take the above gradient step. Which motivates an interesting question: is there some similarity between <u>learning by gradient descent</u> and <u>in-context learning</u>? More formally, is there some relationship between (for some η)

$$\text{Q1.} \qquad T_W(\underline{x}, x_{l+1}, -) \quad \text{and} \quad T_{W+\Delta W}(-) \ ? \tag{2.3}$$

Providing a context is a form of <u>currying</u> where you can generate a new model, and the question is, in part, is this new model a Transformer for some perturbed weights? And if so in what direction?

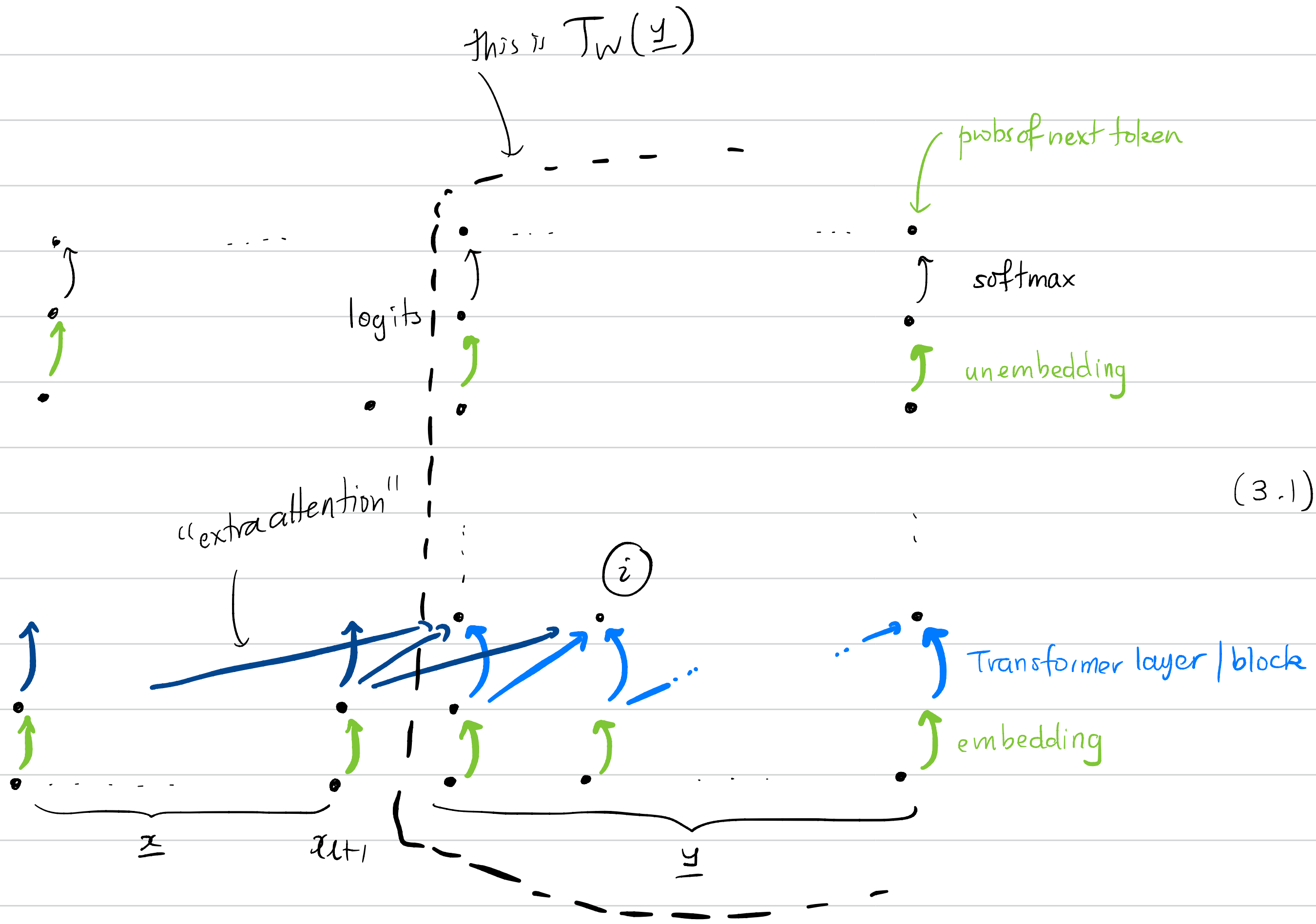
$$\text{Q2.} \qquad T_W(\underline{x}, x_{l+1}, -) \approx T_{W'}(-) \qquad \text{how does } W' \text{ relate to } W?$$

"contexts to weights hypothesis"

Quite unusually for statistical models, the attention weights actually specify an infinite family of models $\{T_W^{(n)}\}_n$ and providing contexts allows us to move between them, potentially <u>internalising many operations such as training</u> which are currently on the outside.

<u>Remark</u> Contra [ICL1, ICL2] there seems little reason to believe in general that $\|W - W'\|$ is small. For instance for the treacherous turn $\underline{x}$ = "In deployment".

We can probably safely say "No" to Q1, except in special cases. We focus on Q2.
Let us consider the algorithm computing $T_W(\underline{x}, x_{l+1}, \underline{y})$



(3.1)

To simplify matters we are going to assume the entities $e_1, \dots, e_{l+1}$ are frozen, and consider the effect on entity i of having $\underline{x}\, x_{l+1}$ in the context.

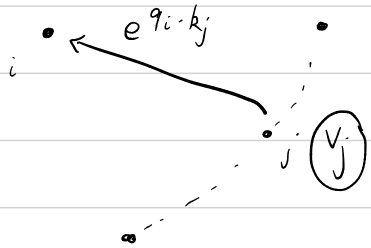
In the attention step the <u>attention weight matrix</u> $W : V \longrightarrow H \oplus H \oplus V$ generates queries, keys and value vectors (notation of [25]) (we ignore biases)

$$(\underbrace{q_i, k_i}_{\text{in } H}, \underbrace{v_i}_{\text{in } V}) := We_i = (W^Q e_i, W^K e_i, W^V e_i)$$

and the update rule is (ignoring layer normalisation and feedforward)

$$e_i' = e_i + \sum_j \mathrm{softmax}(q_i \cdot k_1, \dots, q_i \cdot k_\ell)_j \, v_j$$

$$= e_i + \underbrace{\sum_j \frac{e^{q_i \cdot k_j}}{\sum_u e^{q_i \cdot k_u}} v_j}_{\Delta e_i}$$



If we define softmax(A) for a matrix A column-wise (again following [PH])

$$\mathrm{softmax}(A)_{ij} = \frac{e^{A_{ij}}}{\sum_u e^{A_{uj}}}$$

Then

$$\mathrm{softmax}(K^T Q)_{ji} = \frac{e^{(K^T Q)_{ji}}}{\sum_u e^{(K^T Q)_{ui}}} = \frac{e^{q_i \cdot k_j}}{\sum_u e^{q_i \cdot k_u}}$$

$Q = W^Q E \qquad E[:, i] = e_i$ (ith col)

$K = W^K E \qquad Q[:, i] = q_i$

$$e^{(K^T Q)_{ji}} = e^{q_i \cdot k_j}$$

Hence with $E'[:, i] = e_i'$, (1.2) becomes

$$E' = E + V \,\mathrm{softmax}(K^T Q)$$

$H \leftarrow \mathrm{ent} \leftarrow \mathrm{ent}$

Returning to our situation where entities $e_1, \ldots, e_{l+1}$ are <u>frozen</u>

$$\Delta e_i = \sum_j \frac{e^{q_i \cdot k_j}}{\sum_u e^{q_i \cdot k_u}} v_j$$

$$= \frac{1}{\sum_u e^{q_i \cdot k_u}} \left\{ \sum_{j \text{ frozen}} e^{q_i \cdot k_j} v_j + \sum_{j \text{ not frozen}} e^{q_i \cdot k_j} v_j \right\}$$

<u>Some special cases</u> If $q_i \cdot k_j \approx 0$ for frozen j, and $q_i \cdot k_j$ sufficiently large for other j,

$$\Delta e_i \approx \frac{1}{\sum_{u \text{ not frozen}} e^{q_i \cdot k_u}} \left\{ \sum_{j \text{ not frozen}} e^{q_i \cdot k_j} v_j \right\}$$

"Context is irrelevant"

$$= \text{update in } T_W(\underline{y})$$

Suppose that for each frozen t, there is an unfrozen index $a(t)$ such that $v_j \approx v_{a(j)}$ and write

$$\Delta e_i \approx \frac{1}{\sum_u e^{q_i \cdot k_u}} \left\{ \sum_{j \text{ not frozen}} \left(e^{q_i \cdot k_j} + \sum_{\substack{t \text{ frozen} \\ a(t) = j}} e^{q_i \cdot k_t} \right) v_j \right\}$$

The Log-Sum-Exponential $LSE(x_1, \ldots, x_n) = \log\left(\sum_i e^{x_i}\right)$ is an approximation to $\max\{x_i\}_i$ for x-values large relative to $\log n$. A common approx is $\left(x^* = \max\{x_i\}_i \right)$

$$LSE(x_1, \ldots, x_n) \approx x^* + \log\left(\exp(x_1 - x^*) + \cdots + \exp(x_n - x^*) \right)$$

$$x^* \leq LSE \leq x^* + \log(n)$$

$$e^{q_i \cdot k_j} + \sum_{\substack{t \text{ frozen} \\ a(t) = j}} e^{q_i \cdot k_t} = e^{z}$$

$$z = \mathrm{LSE}\left(\{ q_i \cdot k_j \} \cup \{ q_i \cdot k_t \mid t \text{ frozen } a(t) = j \} \right)$$

$$\approx \quad q_i \cdot k_{j^*} \qquad \text{where this is maximised,}$$

Suppose there is a clear winner, for each unfrozen index j, then

$$\Delta e_i \approx \frac{1}{\sum_u e^{q_i \cdot k_u}} \left\{ \sum_{\substack{j \text{ not} \\ \text{frozen}}} e^{q_i \cdot k_{j^*}} v_j \right\}$$

which is attention but with <u>modified key weights W^K</u> for entity j.
Perhaps sometimes the winner is a frozen entity, and in this way

$$T_W(\underline{x}, x_{\ell+1}, \underline{y}) \approx T_{W'}(\underline{y})$$

where W' includes the modified weights.

<u>Remark</u> • Treat properly using SVD and RG ideas.

In-context learning and SLT

A learning machine in SLT is a tuple (W, p, q, φ) consisting of a parameter space W, a model p, truth q and prior φ. Let us formulate such a learning machine for a large language model like GPT. Note that "GPT" refers to more than just the Transformer model, it also refers to the data distribution and to some extent the training procedure.

Data distribution given some maximum context size C, sample sequences of consecutive tokens of length C from a fixed corpus of tokenstrings (e.g. text)

$$t_1, \dots, t_C \tag{7.1}$$

Then we have input-output pairs

$$\underbrace{(t_1, t_2)}_{1}, (t_1 t_2, t_3), \dots, \underbrace{(t_1 \cdots t_{C-1}, t_C)}_{C-1}. \tag{7.2}$$

The data distribution is the distribution of all examples like (7.2). So if S is the set of possible tokens, we have actually true distributions

$$q_1(x,y) \quad \text{on} \quad S \times S \qquad (\text{prob. of } (t_1, t_2))$$
$$q_2(x,y) \quad \text{on} \quad S^2 \times S \qquad (\text{prob. of } (t_1 t_2, t_3))$$
$$\vdots$$
$$q_{C-1}(x,y) \quad \text{on} \quad S^{C-1} \times S$$

Note that we can make predictions $T_W(t_1), \dots, T_W(t_1 \cdots t_{C-1})$ for all these examples with the same set of weights. The loss function for the Transformer is a sum over the cross-entropies for each of these predictions.

$$L(w) = \sum_{l=1}^{c-1} L^{(l)}(w) \tag{8.1}$$

where for $1 \le l \le c-1$,

$$L^{(l)}(w) = \sum_{(\underline{t}, t') \in S^l \times S} q_l(\underline{t})\, CE\big(q_l(t' | \underline{t}), T_w(\underline{t})\big) \tag{8.2}$$

Thus the true distribution is on $X = \big(\coprod_{l=1}^{c-1} S^l \big) \times S$ and given by $q_1, \dots, q_{c-1}$ taken together. The model is, given $\underline{t} \in S^l$, $t' \in S$,

$$\begin{aligned} p(\underline{t}, t' | w) &= p(t' | \underline{t}, w)\, q_l(\underline{t}) \\ &= T_w(\underline{t})[t']\, q_l(\underline{t}) \end{aligned}$$

and the KL divergence is similarly a sum over l

$$\begin{aligned} K(w) &= \int_X q(\underline{t}, t') \log\left(\frac{q(\underline{t}, t')}{p(\underline{t}, t' | w)} \right) d\underline{t}\, dt' \\ &= \int_X q(\underline{t}, t') \log\left(\frac{q(t' | \underline{t})}{p(t' | \underline{t}, w)} \right) d\underline{t}\, dt' \qquad (8.3) \\ &= \int_X q(t' | \underline{t}) \log\left(\frac{q(t' | \underline{t})}{p(t' | \underline{t}, w)} \right) q(\underline{t})\, d\underline{t}\, dt' \\ &= \sum_{l=1}^{c-1} \int_{S^l \times S} q(t' | \underline{t}) \log\left(\frac{q(t' | \underline{t})}{p(t' | \underline{t}, w)} \right) q_l(\underline{t})\, d\underline{t}\, dt \\ &=: \sum_{l=1}^{c-1} K^{(l)}(w) \end{aligned}$$

where $K^{(l)}$ is the KL divergence for $(\mathcal{W}, p_l, q_l, \varphi)$.

By a familiar calculation

$$K^{(l)}(w) = -\text{Entropy}\, q_l + \int_{S^l} CE\big(q_l(t'|\underline{t}) \,|\, T_w(\underline{t})\big) q_l(\underline{t}) d\underline{t}$$

$$= -\text{Entropy}\, q_l + L^{(l)}(w) \qquad (9.1)$$

so minimising $K^{(l)}(w)$ is the same as minimising $L^{(l)}(w)$.

Remark: Since the same parameter $w \in W$ is used to make next-token predictions for any length this is not a sum of KL divergences for distinct models.

The Bayesian posterior of a language model

The nature of the data distribution (that we see, for a sequence of tokens $x_1, \ldots, x_c$ every initial segment $x_1, \ldots, x_n$) and of the model (we use the same parameters to predict for any sequence length) have some interesting implications for the Bayesian posterior when we incorporate the contexts-to-weights hypothesis.

Recall that given a set of samples $D_n = \{(\underline{t}^{(i)}, t'^{(i)})\}_{i=1}^n$ the posterior is

$$p(w|D_n) = \frac{p(D_n|w)p(w)}{p(D_n)} \qquad (9.2)$$

$$= \frac{1}{Z_n} \varphi(w) \exp(-nL_n(w))$$

where $L_n(w)$ is the empirical loss

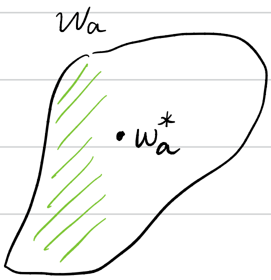
$$\frac{1}{n}\sum_{i=1}^{n} CE\big(q_l(t'^{(i)} | \underline{t}^{(i)}), T_w(\underline{t}^{(i)})\big) \qquad (9.3)$$

$$= -\frac{1}{n}\sum_{i=1}^{n} \log T_w(\underline{t}^{(i)})[t'^{(i)}]$$

The model evidence $Z_n = \int dw\, \varphi(w) \exp(-nL_n(w))$ is the basis of model selection in a Bayesian framework. We view model selection as internalised in LLMs to phase selection where two regions $W_1, W_2 \subseteq W$ are preferred in accordance with which has higher "evidence"

$$Z_n(W_a) := \int_{W_a} dw\, \varphi(w) \exp(-nL_n(w)) \qquad a \in \{1,2\}$$

Or what is the same, which has lower free energy $F_a := -\log Z_n(W_a)$ (the effective Boltzmann weight). Under some conditions we have the



Free Energy Formula

$$Z_n(W_a) \approx nL_n(w_a^*) + \lambda_a \log n$$

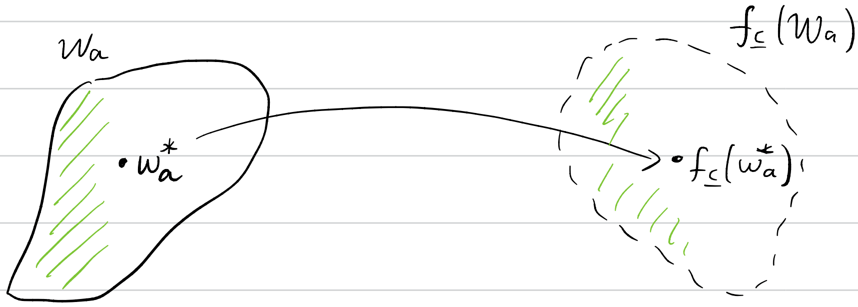
(leading terms in asymptotic exp. see Green book, WBIC paper, DLT3)

Each point $(\underline{t}^{(i)}, t'^{(i)})$ contributes to free energy of every phase.

$$Z_n(W_a) = \int_{W_a} dw\, \varphi(w) \exp\Big(\sum_{i=1}^n \log T_w(\underline{t}^{(i)})[t'^{(i)}] \Big)$$

$$= \int_{W_a} dw\, \varphi(w) \prod_{i=1}^n T_w(\underline{t}^{(i)})[t'^{(i)}]$$

Suppose The contexts-to-weights hypothesis holds for some sample D_n as above, with $\underline{t}^{(i)} = \underline{c}\, \underline{x}^{(i)}$ for all i (in practice we find in D_n some subset like this) in the sense that $T_w(\underline{c}\,\underline{x}^{(i)})[t'^{(i)}] \approx T_{f_{\underline{c}}(w)}(\underline{x}^{(i)})[t'^{(i)}]$ and moreover assume this works for all $w \in W_a$ in some way continuous in w. That is



Note $f_{\underline{c}}$ is likely to be highly degenerate

$$Z_n(W_a, D_n) = \int_{W_a} dw\, \varphi(w) \prod_{i=1}^{n} T_w(\underline{t}^{(i)})[t'^{(i)}]$$

"few shot"

$$= \int_{W_a} dw\, \varphi(w) \prod_{i=1}^{n} T_w(\underline{c}\,\underline{x}^{(i)})[t'^{(i)}]$$

$$\approx \int_{W_a} dw\, \varphi(w) \prod_{i=1}^{n} T_{f_{\underline{c}}(w)}(\underline{x}^{(i)})[t'^{(i)}]$$

$$= \int_{f_{\underline{c}}(W_a)} dw\, \bar{\varphi}(w) \prod_{i=1}^{n} T_w(\underline{x}^{(i)})[t'^{(i)}]$$

absorbs J

$$< Z_n\Big(f_{\underline{c}}(W_a), \big(\underline{x}^{(i)}, t'^{(i)}\big)_{i=1}^{n}\Big)$$

"zero shot"

Since in the asymptotic expansion $f_{\underline{c}}$ contributes additional degeneracy on the LHS. Hence

$$nL_n(w_a^*) + \lambda_a \log n < nL_n(f_{\underline{c}}(w_a^*)) + \lambda_a^{\underline{c}} \log n$$

$\lambda_a^{\underline{c}} = \lambda_a +$ "degeneracy of $f_{\underline{c}}$"

and so (the rough argument goes) $\lambda_a < \lambda_a^{\underline{c}}$. The singularity at w_a^* must be more complex than the one at $f_{\underline{c}}(w_a^*)$ (roughly because $f_{\underline{c}}(w_a^*)$ is "specialised" and this corresponds to $f_{\underline{c}}$ being a projection, i.e. being degenerate. The more specialised the model becomes in context $\underline{c}$, the larger the gap).

Phase as UTM, contexts as codes

Suppose W_a knows to direct predictions to other phases (subroutines) for multiple contexts $\underline{c}_1, \underline{c}_2$. So we suppose a dataset D_n consists of

$$D_n = \{(\underline{c}_1 \underline{x}^{(i)}, y^{(i)})\}_{i=1}^{n_1} \cup \{(\underline{c}_2 \underline{x}^{(j)}, y^{(j)})\}_{j=1}^{n_2} \qquad n_1+n_2=n$$

Then

$$Z_n(W_a, D_n) = \int_{W_a} dw\, \varphi(w) \prod_{i=1}^{n_1} T_w(\underline{c}_1 \underline{x}^{(i)})[y^{(i)}] \cdot \prod_{j=1}^{n_2} T_w(\underline{c}_2 x^{(j)})[y^{(j)}]$$

$$\approx \int_{W_a} dw\, \varphi(w) \prod_{i=1}^{n_1} T_{f_{\underline{c}_1}(w)}(\underline{x}^{(i)})[y^{(i)}] \cdot \prod_{j=1}^{n_2} T_{f_{\underline{c}_2}(w)}(\underline{x}^{(j)})[y^{(j)}]$$

Suppose $f_{\underline{c}_1}$ is constant in all the directions $f_{\underline{c}_2}$ has nonzero partial derivatives at w_a^* and vice-versa, so there are local coordinates u, v in which $f_{\underline{c}_1}(u,v) = f_{\underline{c}_1}(u)$, $f_{\underline{c}_2}(u,v) = f_{\underline{c}_2}(v)$. Then (if this is the only degeneracy)

$$\approx \int_{W_a} du\,dv\, \varphi(u,v) \prod_i T_{f_{\underline{c}_1}(u)}(\cdots) \prod_j T_{f_{\underline{c}_2}(v)}(\cdots)$$

$$= \int du\, \varphi(u) \prod_i T_{f_{\underline{c}_1}(u)}(\cdots) \int dv\, \varphi(v) \prod_j T_{f_{\underline{c}_2}(v)}(\cdots)$$

[W, Remark 2.2]

$$-\log Z_n \approx n L_n(w_a^*) + (\lambda_a^{\underline{c}_1} + \lambda_a^{\underline{c}_2}) \log n$$

We see here a hint of a picture where the true distribution can be approximately divided into subtasks, each of which is much simpler in isolation than the true distribution, and under the context-to-weights hypothesis some contributions to the free energy of the "mother singularity" or phase W_a come from the free energy of submodels ($f_{\leq}(W_a)$ lower-dim) on these tasks.

This suggests an approach to understanding the phase structure of the full model in terms of

(A) Phase transitions where a given context is <u>recognised</u>
(i.e. the context-to-weights hypothesis begins to be applied)

(B) Phase transitions in the submodels

(C) Interactions between the transitions in (B). In the generic case they are simply superimposed.

References

[PH] M. Phuong, M. Hutter "Formal Algorithms for Transformers" 2022.

[GPT3] T. Brown et al "Language models are few shot learners" 2020.

[ICL1] J. van Oswald et al "Transformers learn in-context by gradient descent" 2022

[ICL2] D. Dai et al "Why can GPT learn in context?" 2022.

[2S] J. Clift et al "Logic and the 2-simplicial Transformer" ICLR 2020