Phase Equilibria Transformer: Part 2 - Generative Partitioning

Ali Qajar
ali.qajar @ gmail.com

Abstract

Generative partitioning algorithm is introduced. This model is built on the attentive partitioning algorithm. This extension leverages reaction, an inherently generative process, that enables the model to identify the equilibrium points for generating missing components by looking at the available ones. Introduction of the generative mechanisms to the attentive partitioning algorithm makes the model suited for tasks such as masked language modeling and decoding. The algorithm supports multiple reaction pathways and is linear to the context window. Furthermore, an encoder-decoder architecture called Phase Equilibria Transformer (PET) is introduced. The architecture is a descendant of the original Transformer, is conceptually attention based, uses positional encoding to learn configurations, and produces multiphase outputs per block. As opposed to the Transformer, PET uses partition blocks instead of attention blocks, and embeds the feed-forward into the Partition block, effectively eliminating the need for a separate feed-forward.

Introduction

The goal of this paper is to develop a generative partitioning algorithm. The attentive partitioning algorithm was introduced to replace the Transformer self-attention block for discriminative encoders. The model abandons Query-Key-Value (QKV) scheme, and instead uses a partitioning algorithm which accounts for binary and higher order interactions between the input tokens while keeping the model linear to the context window for structured inputs.

To introduce generative behavior to the partitioning block, one option is to apply masked training techniques like the one done in BERT and make the model learn input structure and predict missing tokens (Devlin et al., 2019). BERT was trained on 15% random token positions. To avoid over-saturating the model with [MASK] token, the target positions were set to [MASK] token 80% of the time, random token 10% of the time and left unchanged 10% of the time. In that, BERT learns to predict the unseen values by leveraging the approximation capacity of the feed-forward blocks.

Another way to induce generative behavior in a partitioning block is to emulate the Transformer decoder stack and use a causal mask (Vaswani et al., 2017). The advantage of the causal mask is that allows to train an input of size N simultaneously, by applying the mask directly to the attention tensor. Note that we can take advantage of this performance boost only for structured masks. For random masking like in BERT we still need to train the network for each masked input separately.

It would be ideal to define a dedicated generative mechanism in the architecture. This has a few advantages: It creates a more structured architecture and allows for better monitoring of the generative pathways. Most importantly, it can reduce computational complexity by merging the feed-forward block into the partition block.

In this paper, generative partitioning algorithm, which is an extension for attention partitioning is introduced. Based on that, Phase Equilibria Transformer (PET), a new Transformer architecture is introduced.

The steps taken in this paper are as follows:

- Incorporate reactions into the attentive partitioning to create generative partitioning.
- Predict reaction parameters using the FF-Partition block.
- Develop a Transformer encoder-decoder model utilizing partition blocks.

Generative Partitioning Algorithm

A reaction is a process through which some components, called reactants, convert to new components, called products. By parameterizing reactions, we gain the ability to forecast the components that will be generated based on the available concentrations of specific components. Assume we have an input made of N tokens. To use a consistent terminology, I refer to the tokens as components. To model component generation, we can define a reaction

$$s_1^{(r)}\hat{z}_1 + s_2^{(r)}\hat{z}_2 + \ldots + s_N^{(r)}\hat{z}_N \Leftrightarrow s_1^{(p)}\hat{z}_1 + s_2^{(p)}\hat{z}_2 + \ldots + s_N^{(p)}\hat{z}_N \quad (1)$$

The bidirectional sign in the middle indicates the reaction is reversible, meaning reactants can convert to products and products can convert back to the reactants. Any reversible reaction, if given enough time, will reach an equilibrium point, at which the rates of forward and backward reactions are equal. Coefficients $s_1 \ldots s_N$ are the stoichiometric coefficients. Superscripts (r) and (p) stand for reactant and product. $z_1, \ldots, z_N$ represent the initial compositions of components $1 \ldots N$. It's important to note that in equation (1) above, we use $\hat{z}_1 \ldots \hat{z}_N$, which is the composition of the system at the equilibrium point. We can write the equation (1) in a compact form

$$\sum s_i^{(r)}\hat{z}_i \Leftrightarrow \sum s_i^{(p)}\hat{z}_i \qquad \text{for } i \in \{1, 2, \ldots, N\} \quad (2)$$

Any reversible reaction has an equilibrium point. The equilibrium state can be found through the law of mass action. It relates the equilibrium compositions of the reactants and products to the equilibrium constant Ω and the stoichiometric coefficients of the reaction, $s_i^{(r)}$ and $s_i^{(p)}$

$$\Omega = \prod_{i=1}^{N} \hat{z}_i^{s_i^{(p)}} \Big/ \prod_{i=1}^{N} \hat{z}_i^{s_i^{(r)}} \quad (3)$$

In reaction equilibrium models, stoichiometric coefficients are used as exponents when calculating the reaction constant. This can be understood by considering that reactions occur through collisions between components. Components with higher composition values have a greater probability of colliding and, consequently, have a more significant contribution to the reaction.

In physical systems, the equilibrium constant is a positive real valued number. Small values $\Omega \ll 1$ indicate the mixture is mostly made of the reactants, whereas large values of the equilibrium constant, $\Omega \gg 1$, indicate the reaction has mostly moved forward to the products side. In our model, we aim to present the equations in a more abstract, mathematical form rather than focusing on their physical interpretation. To achieve this, we make the assumption that both composition and equilibrium constant can take on any real number value, allowing for a more generalized approach to the problem.

Numerically, it is difficult to work with exponents, therefore, let's define $s_i \equiv s_i^{(p)} - s_i^{(r)}$ and writing the reaction equilibrium equation in the log space

$$\sum_{i=1}^{N} s_i log(\hat{z}_i) - log(\Omega) = 0 \tag{5}$$

Before incorporating this model with the attentive partitioning, we need to analyze domains of the variables and parameters. The stoichiometry coefficients are real values $s_i \in R$. Positive stoichiometry means component is produced as a product and negative stoichiometry means the component is consumed as a reactant. The component composition and reaction constant are real values. Since we take their log we need to account for their complex representation. They can't be zero due to the singularity, however are allowed to be negative numbers, therefore their domains are defined as $z_i \in R - \{0\}$ and $\Omega \in R - \{0\}$. This is consistent with what we obtained from the domain analysis in the attentive partitioning paper. Accounting for negative compositions we can write the equilibrium equation (5) as

$$\sum_{i=1}^{N} s_i log(|\hat{z}_i|) - log(\Omega) + \left(\sum_{i=1}^{N} t_i - \tau\right)\pi\, Im = 0 \tag{6a}$$

$$t_i = s_i \quad if\ \hat{z}_i < 0 \quad else\ t_i = 0; \quad \tau = 1 \quad if\ \Omega < 0 \quad else\ \tau = 0 \tag{6b}$$

From equation (6) we have

$$\sum_{i=1}^{N} s_i log(|\hat{z}_i|) - log(|\Omega|) = 0 \tag{7a}$$

$$\sum_{i=1}^{N} t_i - \tau = 0 \tag{7b}$$

$$t_i = s_i \quad if\ \hat{z}_i < 0 \quad else\ t_i = 0, \quad \tau = 1 \quad if\ \Omega < 0 \quad else\ \tau = 0 \tag{7c}$$

With the reaction equilibrium equations prepared, we can now integrate them into the attentive partitioning algorithm. In the original attentive partitioning model, the initial composition was partitioned

into multiple phases. By incorporating reactions into the system, we transform the process into a reactive multiphase system, allowing for the simultaneous consideration of phase partitioning and reactions. The partitioning equations are as follows

$$F_j(L_1, \ldots, L_{P-1}) = \sum_{i=1}^{N} \hat{z}_i(K_{ij} - 1)/u_i = 0 \tag{8a}$$

$$u_i = 1 + \sum_{l=1}^{P-1} L_l(K_{il} - 1) \tag{8b}$$

And the domains of the parameters are

$$z_i \in R - \{0\},\ x_{ij} \in R - \{0\},\ K_{ij} \in R - \{0\},\ L_j \in [0, 1], s_i \in R, \Omega \in R - \{0\}$$

Additionally, the phase compositions are obtained from

$$x_{ip} = \hat{z}_i/u_i \qquad \text{for } i \in \{1, 2, \ldots, N\} \tag{9a}$$

$$K_{ij} \equiv x_{ij}/x_{ip} \qquad \text{for } j \in \{1, 2, \ldots, P-1\} \tag{9b}$$

In order to integrate the phase equations with the reaction equations, we consider the phase equation as the primary objective function, while treating the reaction equations as equality constraints that must be satisfied simultaneously. The nonlinear programming objective is defined as follows

$$minimize\ \ \mathcal{L} \equiv f(z, \hat{z}, K, L, s, \Omega) + \lambda_1 g_1(\hat{z}, s, \Omega) + \lambda_1 g_2(\hat{z}, \Omega) + \mu h(L) \tag{10a}$$

$s.t.$

$$0 \le L_j \le 1 \qquad \text{for } j \in \{1, 2, \ldots, P-1\} \tag{10b}$$

$$h(L) \equiv \left(\sum_{j=1}^{P-1} L_j\right) - 1 \le 0 \tag{10c}$$

$$g_1(\hat{z}, s, \Omega) = \sum_{i=1}^{N} s_i log(|\hat{z}_i|) - log(|\Omega|) = 0 \tag{10d}$$

$$g_2(\hat{z}, \Omega) = \sum_{i=1}^{N} t_i - \tau = 0 \tag{10e}$$

$$t_i = s_i \quad if\ \hat{z}_i < 0 \quad else\ t_i = 0, \quad \tau = 1 \quad if\ \Omega < 0 \quad else\ \tau = 0 \tag{10f}$$

where

$$f(K,\ L)\ =\ RMS(F_j) \equiv \sqrt{\frac{1}{P-1}\sum_{j=1}^{P-1} F_j^{\,2}} \tag{10g}$$

$$L_j = \sigma(\theta_j) \qquad \text{for } j \in \{1,\ 2,\ \ldots,\ P\} \tag{10h}$$

The phase partition objective, as provided in equation (10g), is calculated as the root mean square of the phase functions, given by equation (8). To incorporate the inequality constraint (10c) and the equality constraints (10d) and (10e) into the optimization problem, we employ the method of Lagrange multipliers.

Type I: Reactions in the overall composition: In the previously derived equations (10), we considered a scenario where only a single reaction takes place within the overall composition. However, we can expand this framework to accommodate multiple reactions by introducing the index k. This allows us to define multiple reaction pathways that occur simultaneously within the overall composition. For that we add the index $k \in \{1,\ \ldots,\ R\}$ to the reaction parameters, where R is the number of reactions in the system. Then we have, s_{ik}, Ω_k, t_{ik}, τ_k. The degrees of freedom for Type I system is given by $F = N - P - R$.

Type II: Reactions in all phases with shared stoichiometry: Instead of assuming that reactions are a function of the overall composition, we can consider reactions to be dependent on the phase compositions. We can still employ equation (10) to solve the problem, albeit with the necessary updates to the reaction constraints. It's important to note that while the reaction stoichiometry remains consistent across all phases, the reactions themselves are now a function of the individual phase compositions. The degrees of freedom for Type II system is given by $F = N - P - R$, where R is the number of reactions in the system.

$$\sum_{i=1}^{N} s_{ik} log(|x_{ij}|) - log(|\Omega_k|) = 0 \quad \text{for } k \in \{1,\ 2,\ \ldots,\ R\} \text{ and } j \in \{1,\ 2,\ \ldots,\ P\} \tag{11a}$$

$$\sum_{i=1}^{N} t_{ik} - \tau_k = 0 \tag{11b}$$

$$t_{ik} = s_{ik} \quad if\ x_{ij} < 0 \quad else\ t_{ik} = 0, \qquad \tau_k = 1 \quad if\ \Omega_k < 0 \quad else\ \tau_k = 0 \tag{11c}$$

Type III: Reactions in all phases with independent stoichiometry: Building upon the previous modification, we can further extend the model by allowing each phase to have its own distinct reaction pathways, each with its own independent stoichiometry. This modification makes the process of solving the equations more computationally intensive, as the reactions can be directly obtained from the overall composition, and size of stoichiometry parameters is multiplied by the number of phases. While we can still employ equation (10) for error minimization, it becomes necessary to update the reaction constraints accordingly. The updated reaction constraints will reflect the unique reaction pathways and stoichiometry associated with each individual phase. For that, the parameters from Type II need to be indexed as follows, s_{ijk}, Ω_{jk}, t_{ijk}, τ_{jk}. For Type III system, R_p is the number of reactions per phase, therefore the number of reactions in the system will be $R = P \times R_p$. The degrees of freedom for Type III system is given by $F = N - P \times (R_p + 1)$.

Model Architecture

Figure 1 depicts the generative partitioning block in two stages: (a) during the training phase and (b) during the inference phase. The generative partitioning builds upon the attentive partitioning by incorporating reactions into the model. In the generative partitioning algorithm, the Separator from the attentive partitioning is replaced by a Reactor. The Reactor serves two key functions: during the training phase, it calculates the loss function as defined in equation (10), while during the inference phase, it determines the phase compositions using equation (9). It's important to note that in generative partitioning, the initial composition z cannot be directly used because the reactions cause a shift from the initial composition z to the equilibrium composition $\hat{z}$. As a result, the FF-Partition block is tasked with learning to estimate the equilibrium composition $\hat{z}$ rather than using the initial composition z directly.

The parameters actively participating in each step are represented by the highlighted green bands, while the inactive parameters are denoted by the hashed red bands. During the training phase, all parameters contribute to guiding the FF-Partition in learning the system's features and non-linear characteristics. In this stage, the optimizer works to minimize the loss function, enabling the model to capture the complex relationships within the data. In the inference phase, the phase compositions are efficiently calculated using equation (9) in linear time. During this step, the initial composition, z, and the reaction parameters, namely the stoichiometry, s and the equilibrium constant, Ω are inactive.

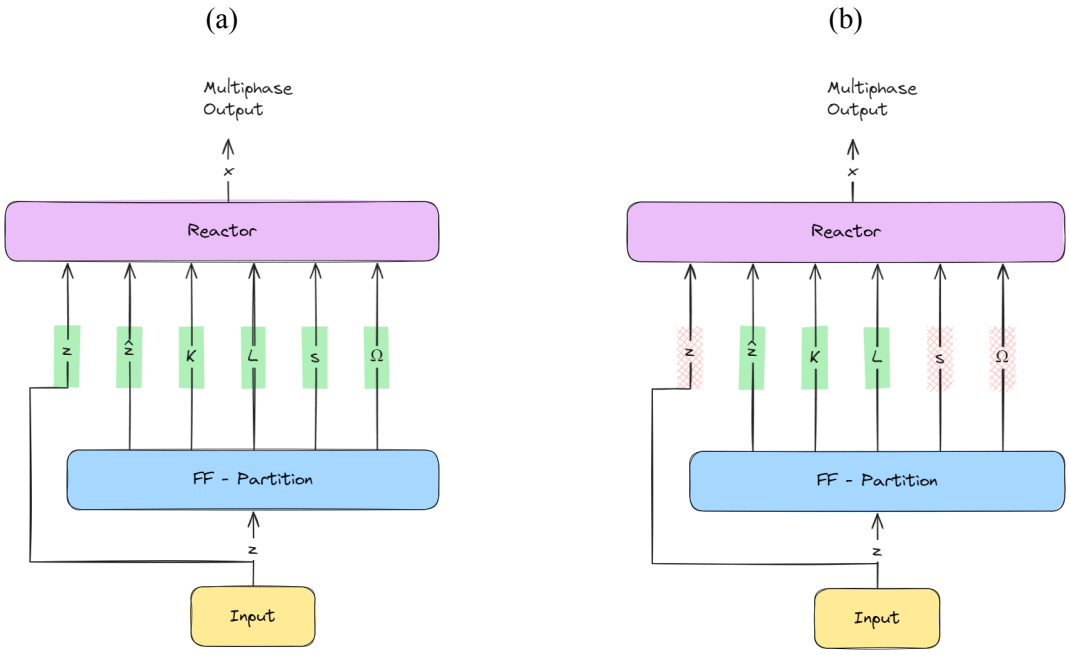

Figure 1: Generative partitioning block during (a) training and (b) inference. At training all input variables and parameters are involved as described by equation 6. At inference, only the variables and phase parameters are involved as described by equation 3.

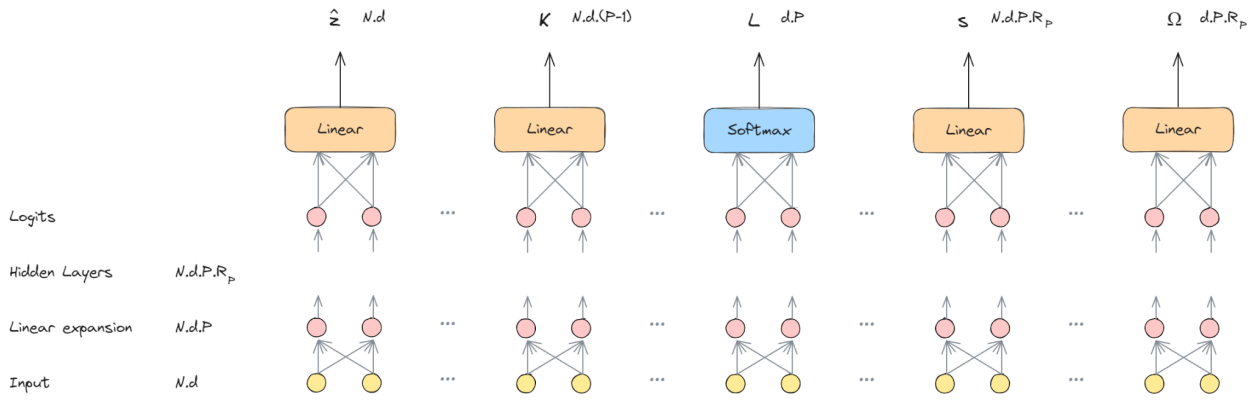

Figure 2: Structure of FF-Partition block.
* Reaction dimensions are given for the Type III generative block.

The FF-Partition block, depicted in Figure 2, is a multi-task learning network that generates five different types of outputs. Its primary objective is to learn the transformation from the initial composition to the equilibrium composition. Additionally, the network learns the partition factors, which encode information about token interactions, akin to attention. Furthermore, it learns to accurately approximate the phase fractions, as these values are utilized during the inference phase.

The input layer is of size $N.d$, where N is the number of tokens and d is the embedding dimension. Since the network has to learn to decompose the overall composition into P phases, the input layer is expanded to linear layer of size $N.d.P$.

The reaction parameters, specifically the stoichiometry and reaction equilibrium constants, are learned during the training process but are not utilized during inference. It's worth noting that the network size undergoes a significant expansion, starting from $N.d$ in the input layer and growing to at least $N.d.P$ in the hidden layers. By providing the network with additional outputs to learn, we supply it with all the necessary information to capture the non-linear relationships within the data, which helps to mitigate overfitting.

The design of the intermediate hidden layers requires a more tailored approach, with their precise dimensions and depth being dictated by the intricacy of the data the network must learn. The suggested approximate dimension of $N.d.P.R_p$ corresponds to a Type III system, in which each phase possesses its own individual set of R_p reactions. It's important to note that both the number of phases P and number of reactions per phase, R_p, serve as tuning parameters that can be adjusted to optimize the network's performance.

The generative partition block is well-suited for masked language models, as it enables the model to uncover intricate patterns and structures within the data. By learning to predict masked or missing tokens, the model can capture complex relationships and dependencies in the input sequences. Furthermore, the generative Partition block is also beneficial for encoder-decoder architectures, which are commonly used in generative tasks.

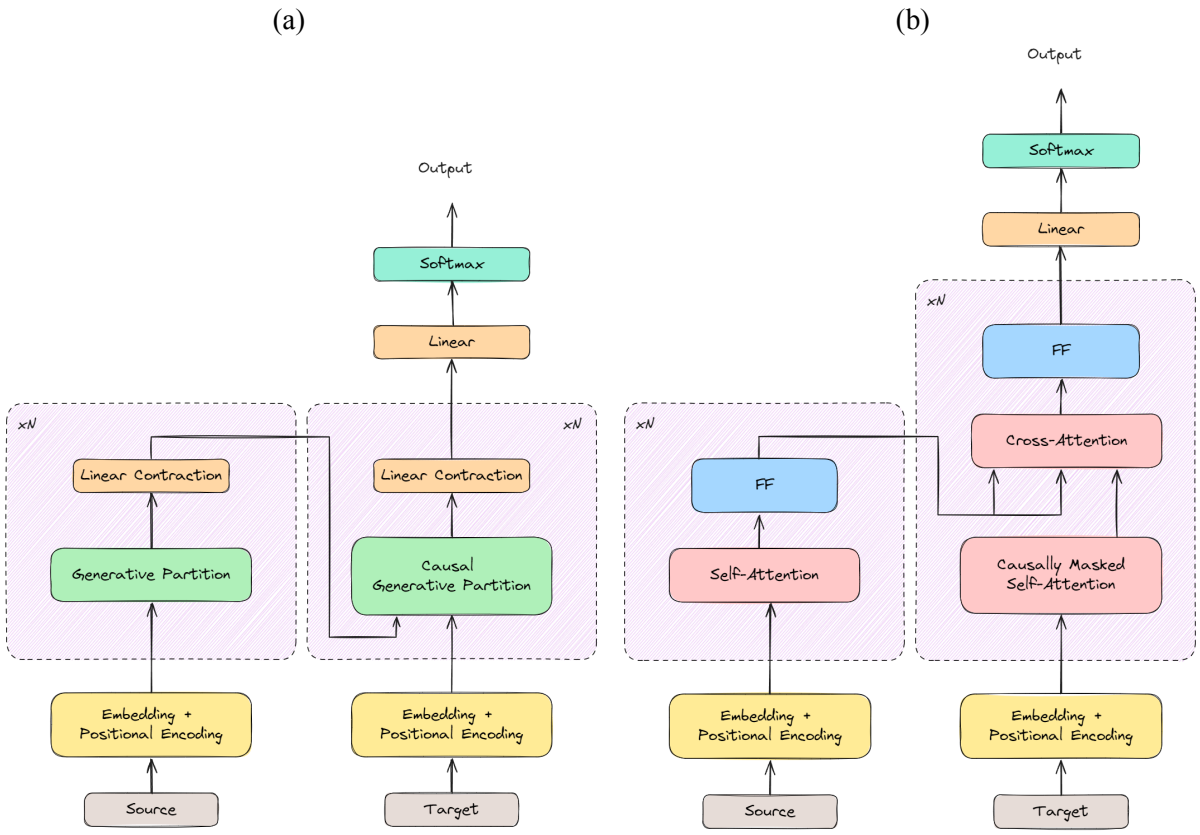

Figure 3: (a) PET encoder-decoder model, (b) Transformer encoder-decoder model. Residuals and layernorms are not shown for brevity.

Figure 3 presents a side-by-side comparison of the encoder-decoder architecture in the Phase Equilibria Transformer (PET) and the traditional Transformer. The key similarities between the two architectures are as follows:

- Both PET and the traditional Transformer are attention-based models, meaning they focus on the compositional aspects rather than the configurational ones. To learn the configurations, they rely on positional and contextual encodings that are embedded into the input.
- In both architectures, multiple heads of filtered outputs are generated. The traditional Transformer refers to this as multihead attention, while PET calls it multihead-multiphase partition. The Transformer produces H outputs corresponding to H attention heads, whereas PET generates $H \times P$ outputs for H partition heads each producing P phases, as illustrated in Figure 4.
- After the attention or partition heads generate their outputs, these outputs are concatenated and reduced in dimensionality. This step ensures that the input size remains consistent across the blocks in both architectures.
- Although not explicitly shown in the figure for simplicity, the use of residual connections and layer normalization is identical in both the traditional Transformer and PET.

- Lastly, the manner in which encoders and decoders are stacked and connected is the same in both the traditional Transformer and PET. While the high-level architectures remain largely similar, they differ in their specific details.

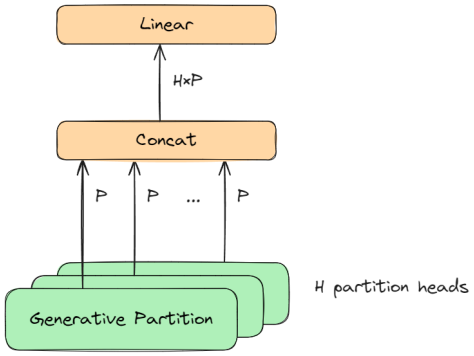

Figure 4: Multihead-Multiphase partition outputs are concatenated and go through linear dimension reduction. Each partition produces P outputs. H partition heads produce $H \times P$ outputs.

As for the differences:

- The traditional Transformer heavily relies on the attention mechanism, employing the Query-Key-Value (QKV) scheme. In this approach, all input tokens interact with each other, and a softmax function later determines which interactions are significant and which ones will be suppressed. However, this limits the model to capturing only pairwise relations between tokens. In contrast, PET redefines this block by replacing it with a partition block. The partition logic treats attention as a dependency and learns it using a feed-forward network. This modification makes the attention buffer tunable, effectively linear in size, and capable of learning interactions of any order. The partition block can be either attentive or generative, depending on the specific use case. Attentive partitioning is well-suited for purely discriminative and classification tasks, while the generative model is appropriate for generative tasks. Following this definition, even for BERT, we should employ generative partitioning instead of attentive partitioning.
- In the traditional Transformer, most, if not all, of the computations related to feature extraction and generation are performed in the feed-forward block that follows each attention block. In contrast, PET incorporates the feed-forward functionality directly into the partition block.
- The Transformer decoder utilizes a masked self-attention mechanism followed by a cross-attention block. PET, on the other hand, achieves the same functionality by employing a causal mixer block followed by a generative partition block.

Figure 5 presents a side-by-side comparison of the PET decoder block and the Transformer decoder block. In the traditional Transformer, each element in the attention matrix is accessible, allowing the model to directly enforce causality on the decoder input. Furthermore, the Transformer utilizes the output from the causal self-attention as the Query vector, while the encoder outputs serve as the Key and Value vectors in the cross-attention block.

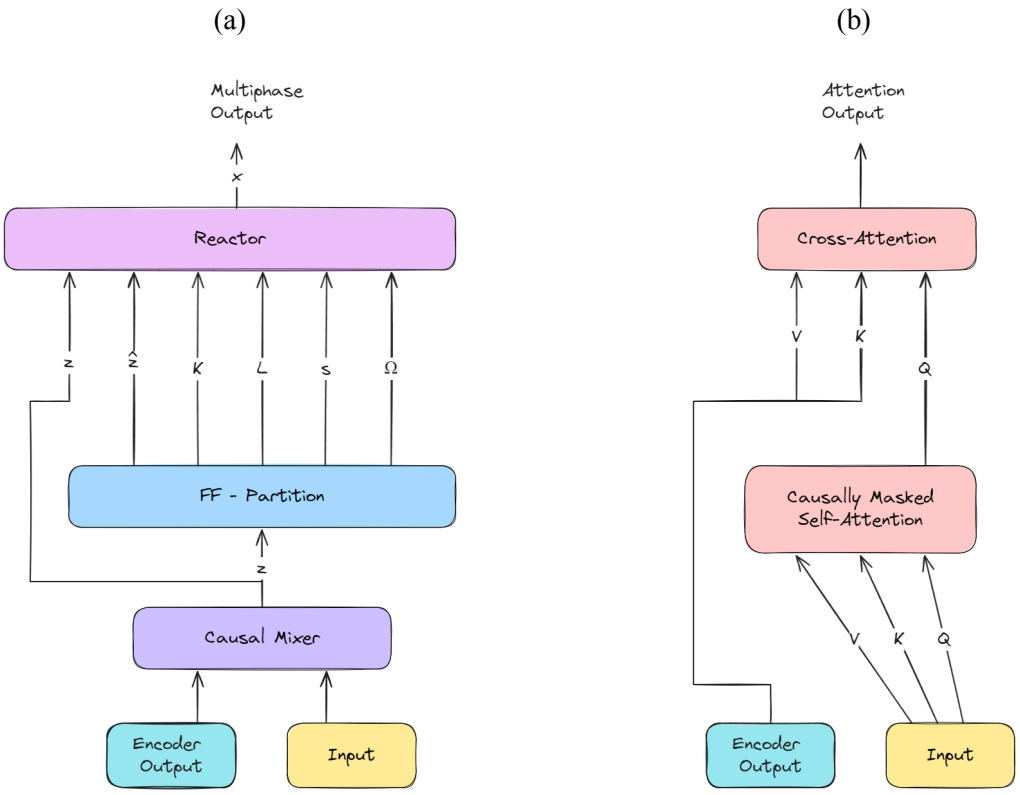

Figure 5: (a) PET decoder block, (b) Transformer decoder block

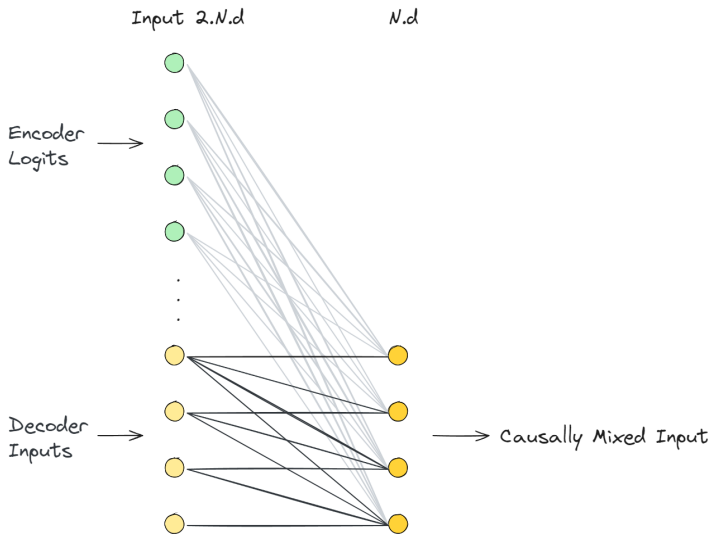

Figure 6: A linear causal mixer block. Decoder side causal mask is embedded in the structure of the network. Decoder inputs can only see the previous decoder inputs in the input sequence. All Encoder Logits are visible to each decoder input.

To enforce causal lookup and enable the decoder to access the encoder output in the partition logic, a causal mixer block is introduced. Figure 6 illustrates a straightforward implementation of the mixer, consisting of a single linear layer. The encoder outputs are made available to all decoder inputs, whereas each decoder token is limited to observing only the previous decoder tokens.

Conclusions

The generative partitioning algorithm introduces a novel approach to modeling complex systems by integrating reactions and phase partitioning. It extends the attentive partitioning model, allowing for the simultaneous consideration of phase and reaction equilibria. The algorithm handles three types of reaction scenarios, offering flexibility in modeling various systems. The generative partition block demonstrates versatility in both masked language models and encoder-decoder architectures.

The Phase Equilibria Transformer (PET) architecture, built upon the generative partitioning model, introduces the partition block and the causal mixer block to the decoder, redefining the attention mechanism and enforcing causality in the decoder. PET's unique structure, which includes multihead-multiphase partition outputs, and the integration of the feed-forward logic within the partition block, allows it to learn higher-order interactions and provides a tunable attention buffer, setting it apart from the traditional Transformer architecture.

References

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, 4171–4186.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention Is All You Need. Advances in Neural Information Processing Systems, 30.