More Identifiable yet Equally Performant Transformers for Text Classification

Rishabh Bhardwaj[1], **Navonil Majumder**[1], **Soujanya Poria**[1], **Eduard Hovy**[2]

[1] Singapore University of Technology and Design, Singapore
[2] Carnegie Mellon University, Pittsburgh, PA, USA
rishabh_bhardwaj@mymail.sutd.edu.sg
{navonil_majumder, sporia}@sutd.edu.sg
hovy@cs.cmu.edu

Abstract

Interpretability is an important aspect of the trustworthiness of a model's predictions. Transformer's predictions are widely explained by the attention weights, i.e., a probability distribution generated at its self-attention unit (head). Current empirical studies provide shreds of evidence that attention weights are not explanations by proving that they are not unique. A recent study showed theoretical justifications to this observation by proving the non-identifiability of attention weights. For a given input to a head and its output, if the attention weights generated in it are unique, we call the weights identifiable. In this work, we provide deeper theoretical analysis and empirical observations on the identifiability of attention weights. Ignored in the previous works, we find the attention weights are more identifiable than we currently perceive by uncovering the hidden role of the key vector. However, the weights are still prone to be non-unique attentions that make them unfit for interpretation. To tackle this issue, we provide a variant of the encoder layer that decouples the relationship between key and value vector and provides identifiable weights up to the desired length of the input. We prove the applicability of such variations by providing empirical justifications on varied text classification tasks. The implementations are available at `https://github.com/declare-lab/identifiable-transformers`.

1 Introduction

Widely adopted Transformer architecture (Vaswani et al., 2017) has obviated the need for sequential processing of the input that is enforced in traditional Recurrent Neural Networks (RNN). As a result, compared to a single-layered LSTM or RNN model, a single-layered Transformer model is computationally more efficient, reflecting in a relatively shorter training time (Vaswani et al., 2017). This advantage encourages the training of deep Transformer-based language models on large-scale datasets. Their learning on large corpora has already attained state-of-the-art (SOTA) performances in many downstream Natural Language Processing (NLP) tasks. A large number of SOTA machine learning systems even beyond NLP (Lu et al., 2019) are inspired by the building blocks of Transformer that is multi-head self-attention (Radford et al., 2018; Devlin et al., 2018).

A model employing an attention-based mechanism generates a probability distribution $\mathbf{a} = \{a_1, \ldots, a_n\}$ over the n input units $z = \{z_1, \ldots, z_n\}$. The idea is to perform a weighted sum of inputs, denoted by $\sum_{i=1}^{n} a_i z_i$, to produce a more context-involved output. The attention vector, $\mathbf{a}$, are commonly interpreted as scores signifying the relative importance of input units. However, counter-intuitively, it is recently observed that the weights generated in the model do not provide meaningful explanations (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019).

Attention weights are (structurally) *identifiable* if we can uniquely determine them from the output of the attention unit (Brunner et al., 2019). Identifiability of the attention weights is critical to the model's prediction to be interpretable and replicable. If the weights are not unique, explanatory insights from them might be misleading.

The *self*-attention transforms an input sequence of vectors $z = \{z_1, \ldots, z_n\}$ to a contextualized output sequence $y = \{y_1, \ldots, y_n\}$, where $y_k = \sum_{i=1}^{n} a_{(k,i)}\, z_i$. The scalar $a_{(k,i)}$ captures how much of the i_{th} token contributes to the contextualization of k_{th} token. A Transformer layer consists of multiple heads, where each head performs self-attention computations, we break the head computations in two phases:

- Phase 1: Calculation of attention weights $a_{(k,i)}$. It involves mapping input tokens to

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 1172–1182
August 1–6, 2021. ©2021 Association for Computational Linguistics

key and query vectors. The dot product of k_{th} query vector and i_{th} key vector gives $a_{(k,i)}$.

- Phase 2: Calculation of a contextualized representation for each token. It involves mapping input tokens to the value vectors. The contextualized representation for k_{th} token can be computed by the weighted average of the value vectors, where the weight of i_{th} token is $a_{(k,i)}$ computed in first phase.

The identifiability in Transformer has been recently studied by Brunner et al. (2019) which provides theoretical claims that under mild conditions of input length, attention weights are not unique to the head's output. Essentially their proof was dedicated to the analysis of the computations in the second phase, i.e., token contextualization. However, the theoretical analysis ignored the crucial first phase where the attention weights are generated. Intrinsic to their analysis, the attention identifiability can be studied by studying only the second phase of head computations. However, even if we find another set of weights from the second phase, it depends on the first phase if those weights can be generated as the part of key-query multiplication.

In this work, we probe the identifiability of attention weights in Transformer from a perspective that was ignored in Brunner et al. (2019). We explore the previously overlooked first phase of self-attention for its contribution to the identifiability in Transformer. During our analysis of the first phase, we uncover the critical constraint imposed by the size of the key vector[1] d_k. The flow of analysis can be described as

- We first show that the attention weights are identifiable for the input sequence length d_s no longer than the size of value vector d_v (§3.1) (Brunner et al., 2019)[2].
- For the case when $d_s > d_v$, we analyse the attention weights as raw dot-product (logits) and the $\mathrm{softmax}$ed dot-product (probability simplex), independently. An important theoretical finding is that both versions are prone to be unidentifiable.
- In the case of attention weights as logits (§3.2.1), we analytically construct another set of attention weights to claim the unidentifiability. In the case of attention weights as $\mathrm{softmax}$ed logits (§3.2.2), we find the attention identifiability to be highly dependent on d_k. Thus, the size of key vector plays an important role in the identifiability of the self-attention head. The pieces of evidence suggest that the current analysis in Brunner et al. (2019) ignored the crucial constraints from the first phase in their analysis.

To resolve the unidentifiability problem, we propose two simple solutions (§4). For the regular setting of the Transformer encoder where d_v depends on the number of attention heads and token embedding dimension, we propose to reduce d_k. This may lead to more identifiable attention weights. Alternatively, as a more concrete solution, we propose to set d_v equal to token embedding dimension while adding head outputs as opposed to the regular approach of concatenation (Vaswani et al., 2017). Embedding dimension can be tuned according to the sequence length up to which identifiability is desired. We evaluate the performance of the proposed variants on varied text classification tasks comprising of ten datasets (§5).

In this paper, our goal is to provide concrete theoretical analysis, experimental observations, and possible simple solutions to identifiability of attention weights in Transformer. The idea behind identifiable variants of the Transformer is—the harder it is to obtain alternative attention weights, the likelier is they are identifiable, which is a desirable property of the architecture. Thus, our contribution are as follows:

- We provide a concrete theoretical analysis of identifiability of attention weights which was missing in the previous work by Brunner et al. (2019).
- We provide Transformer variants that are identifiable and validate them empirically by analysing the numerical rank of the attention matrix generated in the self-attention head of the Transformer encoder. The variants have strong mathematical support and simple to adopt in the standard Transformer settings.
- We provide empirical evaluations on varied text classification tasks that show higher identifiability does not compromise with the task's performance.

[1] The size of key and query vector is expected to be the same due to the subsequent dot product operation

[2] The sequence length denotes number of tokens at input.

2 Background

2.1 Identifiability

A general trend in machine learning research is to mathematically model the input-output relationship from a dataset. This is carried out by quantitatively estimating the set of model parameters that best fit the data. The approach warrants prior (to fitting) examination of the following aspects:

- The sufficiency of the informative data to the estimate model parameters, i.e., practical identifiability. Thus, the limitation comes from the dataset quality or quantity and may lead to ambiguous data interpretations (Raue et al., 2009).
- The possibility that the structure of the model allows its parameters to be uniquely estimated, irrespective of the quality or quantity of the available data. This aspect is called structural identifiability. A model is said to be structurally *unidentifiable* if a different set of parameters yield the same outcome.

In this work, we focus on the structural identifiability (Bellman and Åström, 1970). It is noteworthy that the goodness of the fit of a model on the data does not dictate its structural identifiability. Similar to Brunner et al. (2019), we focus our analysis on the identifiability of attention weights, which are not model parameters, yet demands meaningful interpretations and are crucial to the stability of representations learned by the model.

2.2 Transformer Encoder Layer

We base our analysis on the building block of Transformer, i.e., the encoder layer (Vaswani et al., 2017). The layer has two sub-layers. First sublayer performs the multi-head self-attention, and second is feed-forward network. Given a sequence of tokens $\{x_1, \ldots, x_{d_s}\}$, an embedding layer transforms it to a set of vector $\{z_1, \ldots, z_{d_s}\} \in \mathbb{R}^{d_e}$, where d_e denotes token embedding dimension. To this set, we add vectors encoding positional information of tokens $\{p_1, \ldots, p_{d_s}\} \in \mathbb{R}^{d_e}$.

Multi-head Attention. Input to a head of multi-head self-attention module is $\mathbf{W} \in \mathbb{R}^{d_s \times d_e}$, i.e., a sequence of d_s tokens lying in a d_e-dimensional embedding space. Tokens are projected to d_q-size query, d_k-size key, and d_v-size value vectors using linear layers, resulting in the respective matrices - Query $\mathbf{Q} \in \mathbb{R}^{d_s \times d_q}$, Key $\mathbf{K} \in \mathbb{R}^{d_s \times d_k}$, and Value $\mathbf{V} \in \mathbb{R}^{d_s \times d_v}$. The attention weights $\mathbf{A} \in \mathbb{R}^{d_s \times d_s}$ can be computed by

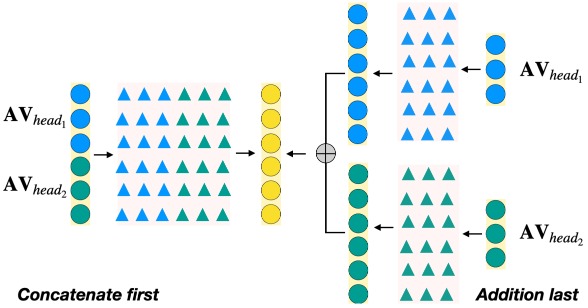

Figure 1: An illustration for a Transformer with two-head attention units. Triangles depict matrix weights. The left side shows concatenation of head outputs fed to a linear layer. The right side shows another interpretation of the same set of operations where we consider a linear transform applied to each head first. The transformed head outputs are then added.

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{d_q}}\right). \tag{1}$$

The $(i,j)_{\text{th}}$ element of $\mathbf{A}$ shows how much of i_{th} token is influenced by j_{th} token. The output of a head $\mathbf{H} \in \mathbb{R}^{d_s \times d_e}$ is given by

$$\mathbf{H} = \mathbf{A}\,\mathbf{V}\,\mathbf{D} = \mathbf{A}\,\mathbf{T}, \tag{2}$$

where $\mathbf{D} \in \mathbb{R}^{d_v \times d_e}$ is a linear layer and the matrix $\mathbf{T} \in \mathbb{R}^{d_s \times d_e}$ denotes the operation $\mathbf{V}\,\mathbf{D}$. The $\mathbb{R}^{d_s \times d_e}$ output of multi-head attention can be expressed as a summation over $\mathbf{H}$ obtained for each head[3]. The i_{th} row of multi-head output matrix corresponds to the d_e dimensional contextualized representation of i_{th} input token. In the original work, Vaswani et al. (2017), the multi-head operation is described as the concatenation of $\mathbf{A}\,\mathbf{V}$ obtained from each head followed by a linear transformation $\mathbf{D} \in \mathbb{R}^{d_e \times d_e}$. Both the explanations are associated with the same sequence of matrix operations as shown in fig. 1.

In regular Transformer setting, a token vector is $t_i \in \{(z_j + p_j)\}_{i=1}^{d_s}$ is d_e = 512 dimensional, number of heads h=8, size of d_k=d_q=d_v=d_e/h=64.

Feed-Forward Network. This sub-layer performs the following transformations on each token representation at the output of a head:

$$y_1 = \text{Linear}_1(\text{Norm}(t_i + \text{head output for } t_i))$$
$$y_2 = \text{Norm}(t_i + \text{ReLU}(\text{Linear}_2(y_1)))$$

Linear_1 and Linear_2 are linear layers with 2048 and 512 nodes, respectively. Norm denotes mini-batch layer normalization.

[3] For simplicity, we have omitted head indices.

3 Identifiability of Attention

The output of an attention head $\mathbf{H}$ is the product of $\mathbf{A}$ and $\mathbf{T}$ (eq. (2)). Formally, we define identifiability of attention in a head:

Definition 3.1. *For an attention head's output $\boldsymbol{H}$, attention weights $\boldsymbol{A}$ are identifiable if there exists a unique solution of $\boldsymbol{A}\,\boldsymbol{T} = \boldsymbol{H}$.*

The above definition can be reformulated as

Definition 3.2. *$\boldsymbol{A}$ is unidentifiable if there exist an $\tilde{\boldsymbol{A}}$, ($\tilde{\boldsymbol{A}} \neq \mathbf{0}$), such that $(\boldsymbol{A}+\tilde{\boldsymbol{A}})$ is obtainable from phase-1 of head computations and satisfy*

$$(\boldsymbol{A}+\tilde{\boldsymbol{A}})\,\boldsymbol{T} = \boldsymbol{A}\,\boldsymbol{T} \implies \tilde{\boldsymbol{A}}\,\boldsymbol{T} = \mathbf{0}. \tag{constraint-R1}$$

Under this constraint, we get $\tilde{a}_i\,\mathbf{T} = 0$ where $\tilde{a}_i$ is the i_{th} row of $\tilde{\mathbf{A}}$. The set of vectors which when multiplied to $\mathbf{T}$ gets mapped to zero describes the left null space of $\mathbf{T}$ denoted by $\mathrm{LN}(\mathbf{T})$. The dimension of the left null space of $\mathbf{T}$ can be obtained by taking the difference of the total number of rows (d_s) and the number of linearly independent rows, i.e, rank of the matrix $\mathbf{T}$ denoted by $\mathrm{rank}(\mathbf{T})$. Let $\dim(\cdot)$ denotes the dimension of a vector space, then

$$\mathrm{LN}(\mathbf{T}) = \{\mathbf{v} \mid \mathbf{v}^T\,\mathbf{T} = \mathbf{0}\} \tag{3}$$
$$\dim\big(\mathrm{LN}(\mathbf{T})\big) = d_s - \mathrm{rank}(\mathbf{T}). \tag{4}$$

3.1 "A" is Identifiable for $d_s \leq d_v$

If $\dim(\mathrm{LN}(\mathbf{T})) = 0$ then $\mathrm{LN}(\mathbf{T}) = \{\mathbf{0}\}$, it leads to the only solution of constraint-R1 that is $\tilde{\mathbf{A}} = \mathbf{0}$. Therefore, the unidentifiabilty condition does not hold. Now we will prove such a situation exists when the number of tokens is not more than the size of value vector.

The matrix $\mathbf{T}$ in eq. (2) is product of $d_s \times d_v$ value matrix $\mathbf{V}$ and $d_v \times d_e$ transformation $\mathbf{D}$. We utilize the fact that the rank of product of two matrices $\mathbf{P}$ and $\mathbf{Q}$ is upper bounded by the minimum of $\mathrm{rank}(\mathbf{P})$ and $\mathrm{rank}(\mathbf{Q})$, i.e., $\mathrm{rank}(\mathbf{P}\,\mathbf{Q}) \leq \min\big(\mathrm{rank}(\mathbf{P}), \mathrm{rank}(\mathbf{Q})\big)$. Thus, the upper bound on $\mathrm{rank}(\mathbf{T})$ in eq. (4) can be determined by

$$\begin{aligned}\mathrm{rank}(\mathbf{T}) &\leq \min\Big(\mathrm{rank}(\mathbf{V}),\ \mathrm{rank}(\mathbf{D})\Big)\\ &\leq \min\Big(\min(d_s, d_v), \min(d_v, d_e)\Big)\\ &\leq \min\Big(d_s, d_v, d_v, d_e\Big)\\ &\leq \min\Big(d_s, d_v\Big) \qquad (\text{as } d_e > d_v)\\ &= \min\Big(d_s, 64\Big)\end{aligned} \tag{5}$$

where the last inequality is obtained for a head in the regular Transformer for which d_v=64.

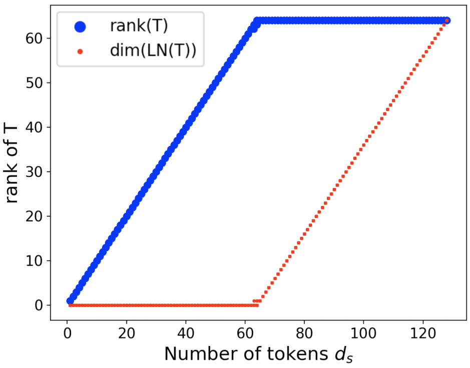

Figure 2: Numerical rank of $\mathbf{T}$ (IMDB) and dimension of its left null space are scattered in blue and red, respectively.

Numerical rank. To substantiate the bounds on $\mathrm{rank}(\mathbf{T})$ as derived above, we set up a model with a single encoder layer (§6). The model is trained to predict the sentiment of IMDB reviews (§5). We feed the review tokens to the model and store the values generated in $\mathbf{T}$ of the first head. A standard technique for calculating the rank of a matrix with floating-point values and computations is to use singular value decomposition. The rank of the matrix will be computed as the number of singular values larger than the predefined threshold[4]. The fig. 2 illustrates how the rank changes with the sequence length d_s. The numerical rank provides experimental support to the theoretical analysis.

$$\mathrm{rank}(\mathbf{T}) = \begin{cases} d_s & \text{if } d_s \leq d_v,\\ d_v & \text{if } d_s > d_v.\end{cases} \tag{6}$$

Thus,

$$\begin{aligned}\dim\big(\mathrm{LN}(\mathbf{T})\big) &= d_s - \mathrm{rank}(\mathbf{T})\\ &= \begin{cases} 0 & \text{if } d_s \leq d_v,\\ (d_s - d_v) & \text{if } d_s > d_v.\end{cases}\\ &= \max\,(d_s - d_v, 0)\end{aligned} \tag{7}$$

With this, we infer $\mathbf{A}$ is identifiable if $d_s \leq d_v = 64$. For the identifiability study, since we focus on a model's capability of learning unique attention weights, we will assume $\mathbf{T}$ has the maximum obtainable rank set by its upper bound.

3.2 Idenitifiability when $d_s > d_v$ (the hidden role of d_k)

In this case, from eq. (7), we obtain a non zero value of $\dim\big(\mathrm{LN}(\mathbf{T})\big)$. It allows us to find infinite $\tilde{\mathbf{A}}$'s satisfying $(\mathbf{A}+\tilde{\mathbf{A}})\,\mathbf{T} = \mathbf{A}\,\mathbf{T}$. However,

[4] The threshold value is $\max(d_s, d_e) * eps * ||\,\mathbf{T}\,||_2$. The *eps* is floating-point machine epsilon value, i.e., 1.19209e-07 in our experiments

constraint-R1 demands $\tilde{\mathbf{A}}$ to be obtainable from the first phase of self-attention. As a first step, we focus our analysis on the attention matrix without applying softmax non-linearity, i.e., $\mathbf{A} = \left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{d_q}}\right)$. The analysis is crucial to identify constraints coming from the first phase of self-attention in Transformer that impact identifiability. Insights from this will help us analyse softmax version of $\mathbf{A}$.

3.2.1 Attention Weights as Logits

Since the logits matrix $\mathbf{A}$ is obtained from the product of $\mathbf{Q}$ and $\mathbf{K}^T$, we can assert that

$$\begin{aligned} \mathrm{rank}(\mathbf{A}) &\leq \min\big(\mathrm{rank}(\mathbf{Q}), \mathrm{rank}(\mathbf{K}^T)\big) \\ &\leq \min\big(d_e, d_k, d_q, d_e\big) \\ &= d_k. \end{aligned} \tag{8}$$

Therefore, the rank of attention matrix producible by the head in the first phase of self-attention can at most be equal to the size of key vectors d_k. On this basis, the head can produce only those $\mathbf{A} + \tilde{\mathbf{A}}$ satisfying

$$\mathrm{rank}(\mathbf{A} + \tilde{\mathbf{A}}) \leq d_k \tag{constraint-R2}$$

Proposition 3.3. *There exists a non-trivial $\tilde{\boldsymbol{A}}$ that satisfy $(\boldsymbol{A} + \tilde{\boldsymbol{A}})\,\boldsymbol{T} = \boldsymbol{A}\,\boldsymbol{T}$ and constraint-R2. Hence, $\boldsymbol{A}$ is unidentifiable.*

Proof. Let $a_1, \ldots, a_{d_s}$ and $\tilde{a}_1, \ldots, \tilde{a}_{d_s}$ denote rows of $\mathbf{A}$ and $\tilde{\mathbf{A}}$, respectively. Without the loss of generality, let $a_1, \ldots, a_{d_k}$ be linearly independent rows. For all $j > d_k$, a_j can be represented as a linear combination $\sum_{i=1}^{d_k} \lambda_i^j a_i$, where λ_i^j is a scalar. Next, we independently choose first k rows of $\tilde{\mathbf{A}}$ that are $\{\tilde{a}_1, \ldots, \tilde{a}_{d_k}\}$ from $\mathrm{LN}(\mathbf{T})$. From the same set of coefficients of linear combination λ_i^j for $i \in \{1, \ldots, d_k\}$ and $j \in \{d_{k+1}, \ldots, d_s\}$, we can construct j_{th} row of $\tilde{\mathbf{A}}$ as $\tilde{a}_j = \sum_{i=1}^{d_k} \lambda_i^j \tilde{a}_i$. Now, since we can construct the j_{th} row of $(\mathbf{A} + \tilde{\mathbf{A}})$ from the linear combination of its first d_k rows as $\sum_{i=1}^{d_k} \lambda_i^j (a_i + \tilde{a}_i)$, the rank of $(\mathbf{A} + \tilde{\mathbf{A}})$ is not more than d_k. For a set of vectors lying in a linear space, a vector formed by their linear combination should also lie in the same space. Thus, the artificially constructed rows of $\tilde{\mathbf{A}}$ belongs to $\mathrm{LN}(\mathbf{T})$. Therefore, there exist an $\tilde{\mathbf{A}}$ that establishes the proposition which claims the unidentifiability of $\mathbf{A}$. □

3.2.2 Attention Weights as Softmaxed Logits

The softmax over attention logits generates attention weights with each row of $\mathbf{A}$ (i.e., a_i's) is constrained to be a probability distribution. Hence, we can define constraint over $\tilde{\mathbf{A}}$ as

$$(\mathbf{A} + \tilde{\mathbf{A}}) \geq \mathbf{0} \tag{P1}$$

$$\tilde{\mathbf{A}}\,\mathbf{T} = \mathbf{0} \tag{P2}$$

$$\tilde{\mathbf{A}}\,\mathbf{1} = \mathbf{0}. \tag{P3}$$

P1 is non-negativity constraint on $(\mathbf{A} + \tilde{\mathbf{A}})$ as it is supposed to be the output of softmax; P2 denotes $\tilde{\mathbf{A}} \in \mathrm{LN}(\mathbf{T})$; P3 can be derived from the fact $(\mathbf{A} + \tilde{\mathbf{A}})\mathbf{1} = \mathbf{1} \implies (\mathbf{A}\,\mathbf{1} + \tilde{\mathbf{A}}\,\mathbf{1}) = \mathbf{1} \implies \tilde{\mathbf{A}}\,\mathbf{1} = \mathbf{0}$ as $(\mathbf{A}\,\mathbf{1} = \mathbf{1})$. Where $\mathbf{1} \in \mathbb{R}^{d_s}$ is the vector of ones. The constraint in P2 and P3 can be combined and reformulated as $\tilde{\mathbf{A}}[\mathbf{T}, \mathbf{1}] = \mathbf{0}$. Following the similar analysis as in eq. (7), we can obtain $\dim\big(\mathrm{LN}([\mathbf{T}, \mathbf{1}])\big) = \max\big(d_s - (d_v + 1), 0\big)$. Disregarding the extreme cases when a_i is a one-hot distribution, Brunner et al. (2019) proved the existence and construction of non-trivial $\tilde{\mathbf{A}}$'s satisfying all the constraints P1, P2, and P3.[5]

However, the proof by Brunner et al. (2019) missed the constraint-R2, hence the existence of a non-trivial $\tilde{\mathbf{A}}$ satisfying only the set of constraints P1, P2 and P3 may not be a valid proposition to claim attention weights unidentifiability. Essentially, the work largely ignored the constraints coming from the rank of the matrix that produces $\mathbf{A}$ after softmax [6]. Let $\mathbf{A}_l$ denote logits $\left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{d_q}}\right)$ and $\mathrm{softmax}(\mathbf{A}_l) = (\mathbf{A} + \tilde{\mathbf{A}})$, where softmax is operated over each row of $\mathbf{A}_l$. We add an extra constraint on $\mathbf{A}_l$

$$\mathrm{rank}(\mathbf{A}_l) \leq d_k. \tag{P4}$$

The constraint P4 confirms if there exists a logit matrix $\mathbf{A}_l$ that can generate $(\mathbf{A} + \tilde{\mathbf{A}})$, given constraints P1, P2, and P3 are satisfied. The possibility of such an $\mathbf{A}_l$ will provide sufficient evidence that $\mathbf{A}$ is unidentifiable. Next, we investigate how the existence of $\tilde{\mathbf{A}}$ is impacted by the size of key vector d_k (query and key vector sizes are the same, i.e., d_q=d_k).

Let $(\mathbf{A} + \tilde{\mathbf{A}})(i, k)$ denotes $(i, k)_{\text{th}}$ element of the matrix. We can retrieve the set of matrices $\mathbf{A}_l$ such that $\mathrm{softmax}(\mathbf{A}_l) = \mathbf{A} + \tilde{\mathbf{A}}$, where

$$\mathbf{A}_l(i, k) = c_i + \log(\mathbf{A} + \tilde{\mathbf{A}})(i, k) \tag{9}$$

[5] For the sake of brevity, we skip the construction method.

[6] (input to the softmax is equivalent to $\mathbf{A}$ in §3.2.1)

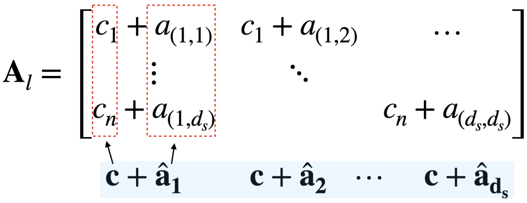

Figure 3: Column vectors $(\boldsymbol{c} + \hat{\mathbf{a}}_{\mathbf{k}})$ of $\mathbf{A}_l$, where $a_{(i,k)}$ represents $\log(\mathbf{A} + \tilde{\mathbf{A}})(i, k)$.

for some arbitrary $c_i \in \mathbb{R}$; log denotes natural logarithm. As shown in fig. 3, the column vectors of $\mathbf{A}_l$ can be written as $\boldsymbol{c} + \hat{\mathbf{a}}_{\mathbf{1}}, \ldots, \boldsymbol{c} + \hat{\mathbf{a}}_{\mathbf{d_s}}$.

For an arbitrarily picked $\tilde{\mathbf{A}}$ satisfying constraint P1, P2, and P3, the dimensions of affine span $\mathcal{S}$ of $\{\hat{\mathbf{a}}_{\mathbf{1}}, \ldots, \hat{\mathbf{a}}_{\mathbf{d_s}}\}$ could be as high as $d_s - 1$ (fig. 4). In such cases, the best one could do is to choose a $\mathbf{c}_{\mathbf{a}} \in \mathcal{S}$ such that the dimension of the linear span of $\{\hat{\mathbf{a}}_{\mathbf{1}} - \mathbf{c}_{\mathbf{a}}, \ldots, \hat{\mathbf{a}}_{\mathbf{d_s}} - \mathbf{c}_{\mathbf{a}}\}$, i.e., $\text{rank}(\mathbf{A}_l)$ is $d_s - 1$. Hence, to satisfy P4, $d_s - 1 \leq d_k \implies d_s \leq d_k + 1$. Thus, the set of $(\mathbf{A} + \tilde{\mathbf{A}})$ satisfying constraint P1, P2 and P3 are not always obtainable from attention head for $d_s > d_k$. We postulate

> Although it is easier to construct $\tilde{\mathbf{A}}$ satisfying constraints P1, P2 and P3, it is hard to construct $\tilde{\mathbf{A}}$ satisfying constraint P4 over the rank of logit matrix $\mathbf{A}_l$. Therefore, $\mathbf{A}$ becomes more identifiable as the size of key vector decreases.

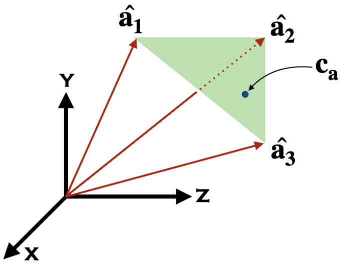

Figure 4: This is a simplified illustration for the case $d_s = 3$. Affine space (translated linear subspace) spanned by vectors $\hat{\mathbf{a}}_{\mathbf{1}}$, $\hat{\mathbf{a}}_{\mathbf{2}}$ and $\hat{\mathbf{a}}_{\mathbf{3}}$. $\mathbf{c}_{\mathbf{a}}$ can be any arbitrary vector in affine space. By putting $\mathbf{c} = -\mathbf{c}_{\mathbf{a}}$, we can obtain a linear subspace whose rank is equal to rank of the affine subspace.

Experimental evidence. We conduct an experiment to validate the minimum possible numerical rank of $\mathbf{A}_l$ by constructing $\tilde{\mathbf{A}}$. For $\tilde{\mathbf{A}}$ to be obtainable from the phase 1, the minimum possible rank of $\mathbf{A}_l$ should not be higher than d_k. From IMDB dataset (§5), we randomly sample a set of reviews with token sequence length d_s ranging from 66 to 128 [7]. For each review, we construct 1000 $\tilde{\mathbf{A}}$'s satisfying constraints P1, P2, and P3 —

First, we train a Transformer encoder-based IMDB review sentiment classifier (§6). We obtain an orthonormal basis for the left null space of $[\mathbf{T}, \mathbf{1}]$ using singular value decomposition. To form an $\tilde{\mathbf{A}}$, we generate d_s random linear combinations of the basis vectors (one for each of its row). Each set of linear combination coefficients is sampled uniformly from $[-10, 10]$. All the rows are then scaled to satisfy the constraint P1 as mentioned in Brunner et al. (2019). Using eq. (9), we obtain a minimum rank matrix $\mathbf{A}_l$'s by putting $\mathbf{c} = -\hat{\mathbf{a}}_{\mathbf{1}}$. Figure 5 depicts the obtained numerical rank of $\mathbf{A}_l$. We observed all the obtained $\mathbf{A}_l$ from $(\mathbf{A} + \tilde{\mathbf{A}})$ (using eq. (9)) are full-row rank matrices. However, from the first phase of self-attention, the maximum obtainable rank of $\mathbf{A}_l$ is $d_k = 64$. Thus, the experimentally constructed $\mathbf{A}_l$'s do not claim unidentifiability of $\mathbf{A}$ as it fails to satisfy the constraint P4, while for Brunner et al. (2019), it falls under the solution set to prove unidentifiability as it meets constraints P1, P2 and P3.

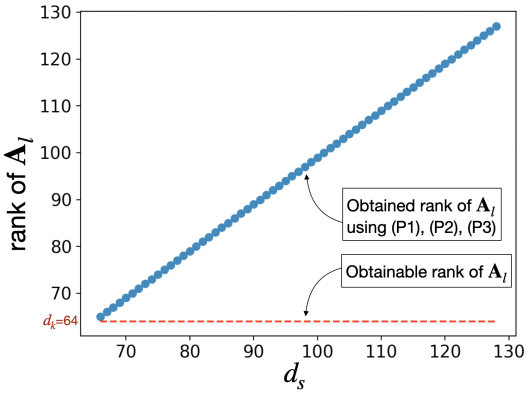

Figure 5: The blue curve denotes the expected rank of $\mathbf{A}_l$'s obtained from $(\mathbf{A} + \tilde{\mathbf{A}})$, where $\tilde{\mathbf{A}}$ satisfies the constraints P1, P2, and P3. The red curve denotes the maximum permissible rank of $\mathbf{A}_l$ that is obtainable from phase 1 of the head.

4 Solutions to Identifiability

Based on the Identifiability analysis in §3, we propose basic solutions to make Transformer's attention weights identifiable.

Decoupling d_k. Contrary to the regular Transformer setting where $d_k = d_v$, a simple approach is to decrease the value of d_k that is the size of the key and query vector. It will reduce the possible

[7] $\dim\left(\text{LN}(\mathbf{T}, \mathbf{1})\right) > 0$ for $d_s > d_v + 1 = 65$

solutions of $\tilde{\mathbf{A}}$ by putting harder constraints on the rank of attention logits, i.e., $\mathbf{A}_l$ in eq. (9). However, theoretically, d_k decides the upper bound on dimensions of the space to which token embeddings are projected before the dot product. Higher the upper bound, more degree of freedom to choose the subspace dimensions as compared to the lower d_k variants. Thus, there is a plausible trade-off when choosing between d_k induced identifiability and the upper bound on the dimension of projected space.

Head Addition. To resolve the unidentifiability issue when sequence length exceeds the size of value vector, we propose to keep the value vector size and token embedding dimension to be more than (or equal to) the maximum allowed input tokens, i.e., $d_v \geq d_{s\text{-max}}$. In Vaswani et al. (2017), d_v was bound to be equal to d_e/h, where d_e is token embedding dimension and h is number of heads. This constraint on d_v is because of the concatenation of h self-attention heads to produce d_e-sized output at the first sub-layer of the encoder. Thus, to decouple d_v from this constraint, we keep $d_v = d_e$ and add each head's output.[8]

5 Classification Tasks

For the empirical analysis of our proposed solutions as mentioned in §4, we conduct our experiments on the following varied text classification tasks:

5.1 Small Scale Datasets

IMDB (Maas et al., 2011). The dataset for the task of **sentiment classification** consist of IMDB movie reviews with their sentiment as positive or negative. Each of the train and test sets contain 25,000 data samples equally distributed in both the sentiment polarities.

TREC (Voorhees and Tice, 2000). We use the 6-class version of the dataset for the task of **question classification** consisting of open-domain, facet-based questions. There are 5,452 and 500 samples for training and testing, respectively.

SST (Socher et al., 2013). Stanford sentiment analysis dataset consist of 11,855 sentences obtained from movie reviews. We use the 3-class version of the dataset for the task of **sentiment classification**. Each review is labeled as positive, neutral, or negative. The provided train/test/valid split is 8,544/2,210/1,101.

5.2 Large Scale Datasets

SNLI (Bowman et al., 2015). The dataset contain 549,367 samples in the training set, 9,842 samples in the validation set, and 9,824 samples in the test set. For the task of recognizing **textual entailment**, each sample consists of a premise-hypothesis sentence pair and a label indicating whether the hypothesis entails the premise, contradicts it, or neutral.

Please refer to Zhang et al. (2015) for more details about the following datasets:

Yelp. We use the large-scale Yelp review dataset for the task of binary **sentiment classification**. There are 560,000 samples for training and 38,000 samples for testing, equally split into positive and negative polarities.

DBPedia. The Ontology dataset for **topic classification** consist of 14 non-overlapping classes each with 40,000 samples for training and 5,000 samples for testing.

Sogou News. The dataset for **news article classification** consist of 450,000 samples for training and 60,000 for testing. Each article is labeled in one of the 5 news categories. The dataset is perfectly balanced.

AG News. The dataset for the **news articles classification** partitioned into four categories. The balanced train and test set consist of 120,000 and 7,600 samples, respectively.

Yahoo! Answers. The balanced dataset for 10-class **topic classification** contain 1,400,000 samples for training and 50,000 samples for testing.

Amazon Reviews. For the task of **sentiment classification**, the dataset contain 3,600,000 samples for training and 400,000 samples for testing. The samples are equally divided into positive and negative sentiment labels.

Except for the SST and SNLI, where the validation split is already provided, we flag 30% of the train set as part of the validation set and the rest 70% were used for model parameter learning.

6 Experimental Setup

Setting up the encoder. We normalize the text by lower casing, removing special characters, etc.[9]

[8] $d_{s\text{-max}} < d_e$ as in the regular Transformer setting.

[9] https://pytorch.org/text/_modules/torchtext/data/utils.html

For each task, we construct separate 1-Gram vocabulary (U) and initialize a trainable randomly sampled token embedding ($U \times d_e$) from $\mathcal{N}(0,1)$. Similarly, we randomly initialize a ($d_{s\text{-max}} \times d_e$) positional embedding.

The encoder (§2.2) takes input a sequence of token vectors ($d_s \times d_e$) with added positional vectors. The input is then projected to key and query vector of size $d_k \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$. For the **regular** Transformer setting, we fix the number of heads h to 8 and the size of value vector $d_v = d_e/h$ that is 64. For each token at the input, the outputs of attention heads are concatenated to generate a d_e-sized vector. For the **identifiable** variant of the Transformer encoder, $d_v = d_e = 512$, this is equal to $d_{s\text{-max}}$ to keep it identifiable up to the maximum permissible number of tokens. The outputs of all the heads are then added. Each token's contextualized representations (added head outputs) are then passed through the feed-forward network (§2.2). For classification, we use the encoder layer's output for the first token and pass it through a linear classification layer. In datasets with more than two classes, the classifier output is $\mathrm{softmax}$ed. In the case of SNLI, we use the shared encoder for both premise and hypothesis; the output of their first tokens is then concatenated just before the final classification layer. We use Adam optimizer, with learning rate =0.001, to minimize the cross-entropy loss between the target and predicted label. For all the experiments, we keep the batch size as 256 and train for 20 epochs. We report the test accuracy obtained at the epoch with the best validation accuracy.

Numerical rank. To generate the numerical rank plot on IMDB dataset as shown in fig. 2, we train a separate Transformer encoder-based classifier. For a particular d_s value, we sample 100 reviews from the dataset with token length $\geq d_s$ and clip each review to the maximum length d_s. The clipping will ensure the number of tokens is d_s before feeding it to the encoder. The numerical rank is calculated for $\mathbf{T}$'s obtained from the first head of the encoder.

7 Results and Discussion

For the identifiable variant, similar to §3.1, we plot the numerical rank of $\mathbf{T}$ with input sequence length as shown in fig. 6. Unlike fig. 2, where $\dim\big(\mathrm{LN}(\mathbf{T})\big)$ linearly increases after $d_s = 64$, we find the dimension is zero for a larger d_s ($\sim$ 380). The zero dimensional (left) null space of $\mathbf{T}$ confirms there exist no nontrivial solution to the constraint constraint-R2, i.e., $\tilde{\mathbf{A}} = \{\mathbf{0}\}$. Thus, the attention weights $\mathbf{A}$ are identifiable for a larger range of length of the input sequence.

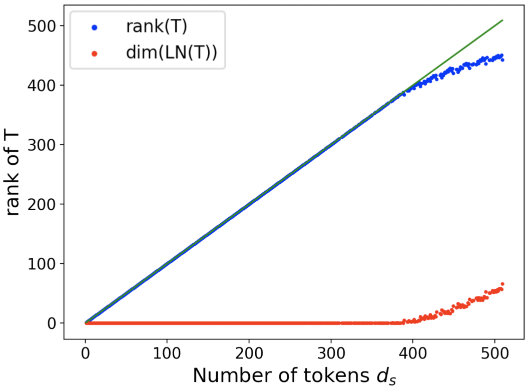

Figure 6: Scatter plots in red and blue show $\mathrm{rank}(\mathbf{T})$ and $\dim\big(\mathrm{LN}(\mathbf{T})\big)$, respectively, for matrices $\mathbf{T}$ obtained from the second phase of attention by feeding IMDB samples to the encoder. The green line shows the desired $\mathrm{rank}(\mathbf{T})$ for which $\dim\big(\mathrm{LN}(\mathbf{T})\big) = 0$ and thus attention weights are identifiable.

It is important that the identifiability of attention weights should not come at the cost of reduced performance of the model. To investigate this issue, we compare the performance of the identifiable Transformer encoder against its regular settings (§6) on varied text classification tasks.

For the regular setting, as discussed in §4 as one of the solutions, the Transformer can be made identifiable by decreasing the size of the key vector d_k. The rows of the Table 1 corresponding to *Con* denotes regular Transformer setting with varying size of key vector. We observe the classification accuracy at the lower d_k is comparable or higher than large d_k values, thus, the enhanced identifiability does not compromise with the model's classification accuracy. However, we notice a general performance decline with an increase in the size of the key vector. We speculate that for simple classification tasks, the lower-dimensional projection for key and query vector works well. However, as the task becomes more involved, a higher dimension for the projected subspace could be essential. Nonetheless, as we do not have strong theoretical findings, we leave this observation for future work.

Another solution to identifiability is to increase d_v to d_e and add the heads' outputs. This setting corresponds to the *Add* rows in the Table 1. For key vector size d_k= 1, 2, and 4, We find the identifiable Transformer's performance is comparable

Dataset	Version	Size of key vector (d_k)								
		1	2	4	8	16	32	64	128	256
IMDB	*Con*	0.884	0.888	0.886	0.888	0.846	0.824	0.803	0.788	0.755
	Add	0.888	0.885	0.887	0.884	0.886	0.882	0.877	0.832	0.825
TREC	*Con*	0.836	0.836	0.840	0.822	0.823	0.764	0.786	0.706	0.737
	Add	0.841	0.842	0.835	0.842	0.841	0.836	0.809	0.809	0.771
SST	*Con*	0.643	0.625	0.627	0.609	0.603	0.582	0.574	0.573	0.554
	Add	0.599	0.618	0.628	0.633	0.628	0.629	0.592	0.581	0.586
SNLI	*Con*	0.675	0.674	0.673	0.672	0.662	0.659	0.659	0.655	0.648
	Add	0.683	0.677	0.674	0.676	0.673	0.669	0.663	0.664	0.655
Yelp	*Con*	0.913	0.911	0.907	0.898	0.879	0.862	0.857	0.849	0.837
	Add	0.914	0.915	0.916	0.914	0.915	0.916	0.910	0.909	0.891
DBPedia	*Con*	0.979	0.977	0.977	0.971	0.966	0.961	0.957	0.951	0.949
	Add	0.979	0.978	0.979	0.977	0.978	0.973	0.970	0.969	0.964
Sogou	*Con*	0.915	0.907	0.898	0.900	0.893	0.888	0.868	0.858	0.838
	Add	0.915	0.908	0.906	0.904	0.913	0.914	0.910	0.906	0.899
AG News	*Con*	0.906	0.903	0.904	0.904	0.886	0.877	0.870	0.870	0.869
	Add	0.902	0.908	0.907	0.906	0.897	0.899	0.901	0.897	0.893
Yahoo	*Con*	0.695	0.690	0.684	0.664	0.644	0.627	0.616	0.597	0.574
	Add	0.697	0.695	0.696	0.693	0.693	0.694	0.688	0.649	0.683
Amazon	*Con*	0.924	0.925	0.923	0.922	0.900	0.892	0.887	0.882	0.873
	Add	0.925	0.923	0.925	0.924	0.924	0.920	0.907	0.896	0.889

Table 1: The test accuracy on varied text classification tasks spread over ten datasets. *Con* means the regular concatenation of heads with $d_v = d_e/h$, *Add* denotes encoder variant where $d_v = d_e$ and outputs of heads are added. In the regular Transformer encoder Con, the concatenation of d_v-sized output of h heads followed by $d_e \times d_e$ linear transformation can be understood as first doing linear $d_v \times d_e$ linear transform of each head and then addition of the transformed output (fig. 1). In the *Add* variant, we first add h d_v-sized head outputs followed by $d_e \times d_e$ linear transformation.

to the regular settings. For $d_k \geq 8$, as a general observation, we find the performance of *Add* does not drop as drastically as *Con* with an increase in d_k. This could be due to the larger size of value vector leading to the more number of parameters in *Add* that compensate for the significant reduction in the model's accuracy.

On the large-scale datasets, we observe that *Add* performs slightly better than *Con*. Intuitively, as shown in fig. 1, we can increase the size of value vector to increase the dimension of the space on which each token is projected. A higher dimensional subspace can contain more semantic information to perform the specific task.

Even though the theoretical analysis shows the possibility of a full row rank of $\mathbf{T}$ and identifiable attention weights, the $\mathbf{T}$ obtained from a trained model might not contain all the rows linearly independent as d_s increases. We can explain this from the semantic similarities between words co-occurring together (Harris, 1954). The similarity is captured as the semantic relationship, such as dot product, between vectors in a linear space. As the number of tokens in a sentence, i.e., d_s increases, it becomes more likely to obtain a token vector from the linear combination of other tokens.

8 Conclusion

This work probed Transformer for identifiability of self-attention, i.e., the attention weights can be uniquely identified from the head's output. With theoretical analysis and supporting empirical evidence, we were able to identify the limitations of the existing study by Brunner et al. (2019). We found the study largely ignored the constraint coming from the first phase of self-attention in the encoder, i.e., the size of the key vector. Later, we proved how we can utilize d_k to make the attention weights more identifiable. To give a more concrete solution, we propose encoder variants that are more identifiable, theoretically as well as experimentally, for a large range of input sequence lengths. The identifiable variants do not show any performance drop when experiments are done on varied text classification tasks. Future works may analyse the critical impact of identifiability on the explainability and interpretability of the Transformer.

Acknowledgments

This research is supported by A*STAR under its RIE 2020 Advanced Manufacturing and Engineering programmatic grant, Award No.– A19E2b0098.

References

Ror Bellman and Karl Johan Åström. 1970. On structural identifiability. *Mathematical biosciences*, 7(3-4):329–339.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. *arXiv preprint arXiv:1908.04211*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Andreas Raue, Clemens Kreutz, Thomas Maiwald, Julie Bachmann, Marcel Schilling, Ursula Klingmüller, and Jens Timmer. 2009. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics*, 25(15):1923–1929.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*.

A Background on Matrices

A.1 Span, Column space and Row space

Given a set of vectors $\mathbf{V} := \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, the span of **V**, $\mathrm{span}(\mathbf{V})$, is defined as the set obtained from all the possible linear combination of vectors in **V**, i.e.,

$$\mathrm{span}(\mathbf{V}) := \{\sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mid \lambda_i \in \mathbb{R},\ i \in \{1, 2, \ldots, n\}\}.$$

The $\mathrm{span}(\mathbf{V})$ can also be seen as the smallest vector space that contains the set **V**.

Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the column space of **A**, $\mathrm{Cs}(\mathbf{A})$, is defined as space spanned by its column vectors. Similarly, the row space of **A**, $\mathrm{Rs}(\mathbf{A})$, is the space spanned by the row vectors of **A**. $\mathrm{Cs}(\mathbf{A})$ and $\mathrm{Rs}(\mathbf{A})$ are the subspaces of the real spaces $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively. If the row vectors of **A** are linearly independent, the $\mathrm{Rs}(\mathbf{A})$ will span $\mathbb{R}^m$. A similar argument holds between $\mathrm{Cs}(\mathbf{A})$ and $\mathbb{R}^n$.

A.2 Matrix Rank

The rank of a matrix **P** (denoted as $\mathrm{rank}(\mathbf{P})$) tells about the dimensions of the space spanned by the row vectors or column vectors. It can also be seen as the number of linearly independent rows or columns. The following properties hold

$$\mathrm{rank}\left(\mathbf{P}\right) \leq \min\Big(m_p, n_p\Big)$$
$$\mathrm{rank}\left(\mathbf{P}\,\mathbf{Q}\right) \leq \min\Big(\mathrm{rank}(\mathbf{P}), \mathrm{rank}(\mathbf{Q})\Big).$$

Where, **P** and **Q** are $m_p \times n_p$ and $m_q \times n_q$ dimensional matrices, respectively.

A.3 Null Space

The left null space of a $m_p \times n_p$ matrix **P** can be defined as the set of vectors **v** -

$$\mathrm{LN}\left(\mathbf{P}\right) = \{\mathbf{v}^T \in \mathbb{R}^{1 \times m_p} \mid \mathbf{v}^T\,\mathbf{P} = 0\} \qquad (10)$$

If the rows of **P** are linearly independent (**P** is full-row rank) the left null space of **P** is zero dimensional. The only solution to the system of equations $\mathbf{v}\,\mathbf{P} = 0$ is trivial, i.e., **v**=0. The dimensions of the null space, known as nullity, of **P** can be calculated as

$$\dim\left(\mathrm{LN}(\mathbf{P})\right) = m_p - \mathrm{rank}(\mathbf{P}). \qquad (11)$$

The nullity of **P** sets the dimensions of the space **v** lies in. In §3, we utilize our knowledge of appendix A.2 and appendix A.3 to analyse identifiability in a Transformer.