ON IDENTIFIABILITY IN TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work we contribute towards a deeper understanding of the Transformer architecture by investigating two of its core components: self-attention and contextual embeddings. In particular, we study the identifiability of attention weights and token embeddings, and the aggregation of context into hidden tokens. We show that attention weights are not unique and propose *effective attention* as an alternative for better interpretability. Furthermore, we show that input tokens retain their identity in the first hidden layers and then progressively become less identifiable. We also provide evidence for the role of non-linear activations in preserving token identity. Finally, we demonstrate strong mixing of input information in the generation of contextual embeddings by means of a novel quantification method based on gradient attribution. Overall, we show that self-attention distributions are not directly interpretable and present tools to further investigate Transformer models.

1 INTRODUCTION

In this paper we investigate neural models of language based on self-attention by leveraging the concept of *identifiability*. Intuitively, identifiability refers to the ability of a model to learn stable representations. This is arguably a desirable property, as it affects the replicability and interpretability of the model's predictions. Concretely, we focus on two aspects of identifiability. The first is related to *structural identifiability* (Bellman & Åström, 1970): the theoretical possibility (a priori) to learn a unique optimal parameterization of a statistical model. From this perspective, we analyze the identifiability of attention weights, what we call *attention identifiability*, in the self-attention components of transformers (Vaswani et al., 2017), one of the most popular neural architectures for language encoding and decoding. We also investigate *token identifiability* as the fine-grained, word-level mappings between input and output generated by the model. The role of attention as a means of recovering input-output mappings, and various types of explanatory insights, is currently the focus of much research and depends to a significant extent on both types of identifiability.

We contribute the following findings to the ongoing work: With respect to attention indentifiability, in Section 3 we show that – under mild conditions with respect to input sequence length and attention head dimension – the attention weights for a given input are not identifiable. This implies that there can be many different attention weights that yield the same output. This finding challenges the direct interpretability of attention distributions. As an alternative, we propose the concept of *effective attention*, a diagnostic tool that examines attention weights for model explanations by removing the weight components that do not influence the model's predictions.

With respect to token identifiability, in Section 4, we devise an experimental setting where we probe the hypothesis that contextual word embeddings maintain their identity as they pass through successive layers of a transformer. This is an assumption made in much current research, which has not received a clear validation yet. We find that this assumption is correct in earlier layers, while it does not always hold in later layers. We propose a partial explanation for the identifiability of tokens, supported by empirical evidence, according to which the non-linearities in the feedforward layers are responsible to a large degree for maintaining token identity.

In Section 5 we further investigate the contribution of all input tokens in the generation of the contextual embeddings in order to quantify the mixing of token and context information. For this purpose, we introduce *Hidden Token Attribution*, a quantification method based on gradient attribution. We find that self-attention strongly mixes context and token contributions. The token contribution decreases monotonically with depth, but the corresponding token typically remains the largest individual contributor. We also find that, despite visible effects of long term dependencies, the context aggregated into the hidden embeddings is mostly local. We notice how, remarkably, this must be an effect of learning.

2 Background on Transformers

The Transformer (Vaswani et al., 2017) is the neural architecture of choice for natural language processing (NLP). At its core it consists of several multi-head self-attention layers. In these, every token of the input sequence attends to all other tokens by projecting its embedding to a query, key and value vector. Formally, let $Q \in \mathbb{R}^{d_s \times d_q}$ be the query matrix, $K \in \mathbb{R}^{d_s \times d_q}$ the key matrix and $V \in \mathbb{R}^{d_s \times d_v}$ the value matrix, where d_s is the sequence length and d_q and d_v the dimension of the query and the value vectors, respectively. The output of an attention head is given by:

$$\text{Attention}(Q, K, V) = A \cdot V \qquad \text{with} \quad A = \text{softmax}\left(\frac{QK^T}{\sqrt{d_q}}\right) \tag{1}$$

The attention matrix $A \in \mathbb{R}^{d_s \times d_s}$ calculates for each token in the sequence how much the hidden embedding at this sequence position attends to each of the other (hidden) embeddings. Self-attention is a non-local operator, which means that at any layer a token can attend to all other tokens regardless of the distance in the input. Self-attention thus produces so-called *contextual word embeddings*, as successive layers gradually aggregate contextual information into the embedding of the input word.

We focus on a Transformer model called BERT (Devlin et al., 2019), although our analysis can be easily extended to other models such as GPT (Radford et al., 2018; 2019). BERT operates on input sequences of length d_s. We denote input tokens in the sentence as $\boldsymbol{x}_i$, where $i \in [1, ..., d_s]$. We use $\boldsymbol{x}_i \in \mathbb{R}^d$ with embedding dimension d to refer to the sum of the token-, segment- and position embeddings corresponding to the input word at position i. We denote the contextual embedding at position i and layer l as $\boldsymbol{e}_i^l$. Lastly, we refer to the inputs and embeddings of all sequence positions as matrices X and E, respectively, both in $\mathbb{R}^{d_s \times d}$. For all experiments we use the pre-trained uncased BERT-Base model as provided by Devlin et al. (2019)[1].

3 Attention Identifiability

In this section, we present the identifiability analysis of self-attention weights. Drawing an analogy with structural identifiability (Bellman & Åström, 1970), we state that the attention weights of a attention head for a given input are *identifiable* if they can be uniquely determined from the head's output.[2] We emphasize that attention weights are input dependent and *not model parameters*. However, their identifiability affects the interpretability of the output, as discussed in (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). If attention is not identifiable, explanations based on attention may be unwarranted.

The output of a multi-head attention layer is the summation of the output of h single heads (cf. Eq. 1) multiplied by the matrix $H \in \mathbb{R}^{d_v \times d}$ with reduced head dimension $d_v = d/h$,

$$\text{Attention}(Q, K, V)H = AEW^V H = AT \tag{2}$$

where $W^V \in \mathbb{R}^{d \times d_v}$ projects the embedding E into the value matrix $V = EW^V$, and we define $T = EW^V H$. Here, the layer and head indices are omitted for simplicity, since the proof below is valid for each individual head and layer in Transformer models. We now prove, by analyzing the null space dimension of T, that attention weights are not identifiable using the head or final model output.

[1]`https://github.com/google-research/bert`

[2]Cf. Appendix A.1 for some background on attention indentifiability.

3.1 Upper Bound for rank(T)

We first derive the upper bound of the rank of matrix $T = EW^VH$. Note that rank$(ABC) \leq \min(\text{rank}(A), \text{rank}(B), \text{rank}(C))$, therefore,

$$\begin{aligned} \text{rank}\,(T) &\leq \min\big(\text{rank}(E), \text{rank}(W^V), \text{rank}(H)\big) \\ &\leq \min(d_s, d, d, d_v, d_v, d) \\ &= \min\,(d_s, d_v)\,. \end{aligned} \tag{3}$$

The second step holds because rank$(E) \leq \min(d_s, d)$, rank$(W^V) \leq \min(d, d_v)$ and rank$(H) \leq \min(d_v, d)$.

3.2 The null space of T

The null space of T describes all vectors that are mapped to 0 by T:

$$\text{null}(T) = \{\tilde{x}^T \in \mathbb{R}^{1\times d_s} | \tilde{x}^T T = 0\} \tag{4}$$

Its special property is that, for $\tilde{A} = [\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_{d_s}]^T$ where $\tilde{x}_i^T$ is any vector in this null space,

$$(A + \tilde{A})T = AT. \tag{5}$$

This implies that there are infinitely many different attention weights $A + \tilde{A}$ that lead to the exact same attention layer output and final model outputs, if the dimension of the null space is not zero.

Due to the Rank Nullity theorem, the dimension of the null space of T is

$$\dim(\text{null}(T)) = d_s - \text{rank}\,(T) \geq d_s - \min\,(d_s, d_v) = \begin{cases} d_s - d_v, & \text{if } d_s > d_v \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where equality holds if E, W^V and H are of full rank and their matrix product does not bring further rank reductions. Hence, when the sequence length is larger than the attention head dimension ($d_s > d_v$), self-attention is non-identifiable. Furthermore, the null space dimension increases with the sequence length.

3.3 Effective attention

The non-identifiability of multi-head self-attention, due to the existence of the non-trivial null space of T, challenges the interpretability of attention weights. However, one can decompose attention weights A into the component in the null space $A^{\parallel}$ and the component orthogonal to the null space $A^{\perp}$:

$$AT = (A^{\parallel} + A^{\perp})T = A^{\perp}T \tag{7}$$

since $A^{\parallel} \in \text{null}(T) \implies A^{\parallel}T = 0$. Hence, we propose a novel concept named *effective attention*,

$$A^{\perp} = A - \text{Projection}_{\text{null}(T)}A, \tag{8}$$

which is the part of the attention weights that actually influence model behavior.

Effective attention can serve as a better diagnostic tool for examining attention weights. To illustrate the point, Figure 1 compares (a) raw attention A and (b) effective attention $A^{\perp}$. Using the same Wikipedia samples as in Clark et al. (2019) with maximum sequence length 128, we compute the average attention of BERT and compare it to the corresponding average effective attention. Clark et al. (2019) conclude that the [CLS] token attracts more attention in early layers, the [SEP] tokens attract more in middle layers, and periods and commas do so in deep layers. However, effective attention weights suggest a different interpretation: while periods and commas seem to generally attract more attention than [CLS] and [SEP], the pattern observed by Clark et al. (2019) has disappeared. An additional example showing similar results can be found in Appendix A.2.

We note that the Pearson correlation between effective and raw attention decreases with sequence length as shown in Figure 1c. This is in line with our theoretical finding in Eq. 6 that states an

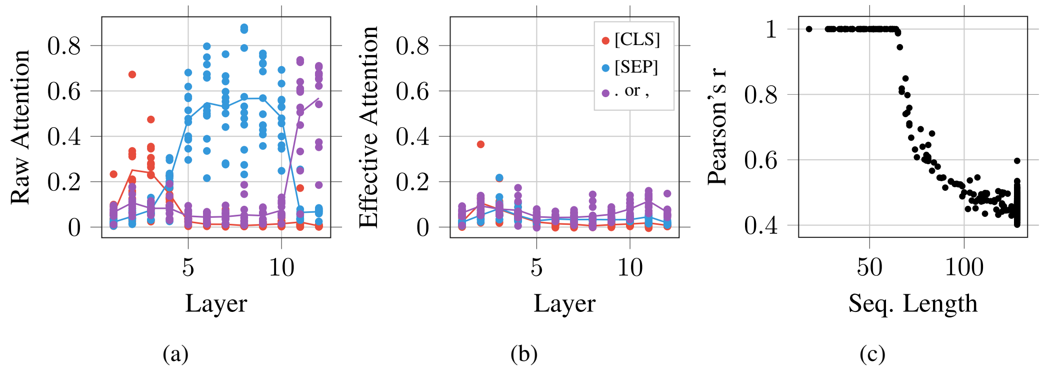


Figure 1: (a) Raw attention vs. (b) effective attention, where each point represents the average (effective) attention of a given head to a token type. (c) Each point represents the Pearson correlation coefficient of effective attention and raw attention as a function of token length.

increase in the dimension of the null space with the sequence length. Given a bigger null space, more of the raw attention becomes irrelevant, yielding a lower correlation between effective and raw attention. Note also that for sequences with fewer than $d_v = 64$ tokens, the associated null space dimension is zero, and hence attention and effective attention are identical (Pearson correlation of value 1). This loss of correlation with increased sequence length questions the use of attention as explanation in practical models, where it is not uncommon to use large sequence lengths. A few examples include: BERT for question answering (Alberti et al., 2019) and XL-Net (Yang et al., 2019) with $d_s = 512$, or document translation (Junczys-Dowmunt, 2019) with $d_s = 1000$.

4 Token Identifiability

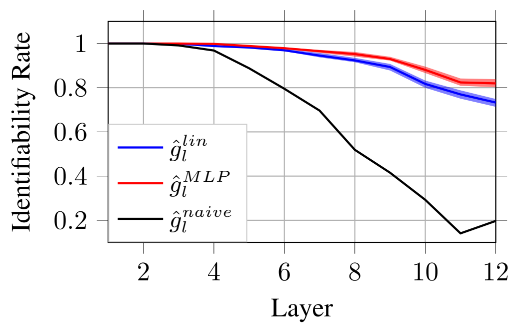
We now study the other fundamental element of transformers; the internal vector representations of tokens, or contextual word embeddings. It is commonly assumed that a contextual word embedding keeps its "identity", which is tied to the input word, as it passes through the self-attention layers. Specifically, we identify three cases where this assumption is made implicitly without justification.

First, visualizations/interpretations linking attention weights to attention between words, when in fact the attention is between embeddings, i.e., mixtures of multiple words (Vaswani et al., 2017; Devlin et al., 2019; Vig, 2019; Clark et al., 2019; Raganato & Tiedemann, 2018; Voita et al., 2019; Tang et al., 2018; Wangperawong, 2018; Padigela et al., 2019; Baan et al., 2019; Dehghani et al., 2019; Zenkel et al., 2019). Second, accumulation methods that sum the attention to a specific sequence position over layers and/or attention heads, when the given position might encode a different mixture of inputs in each layer (Clark et al., 2019; Baan et al., 2019; Klein & Nabi, 2019; Coenen et al., 2019). Finally, using classifiers to probe hidden embeddings for word-specific aspects without factoring in how much the word is still represented (Lin et al., 2019; Peters et al., 2018).

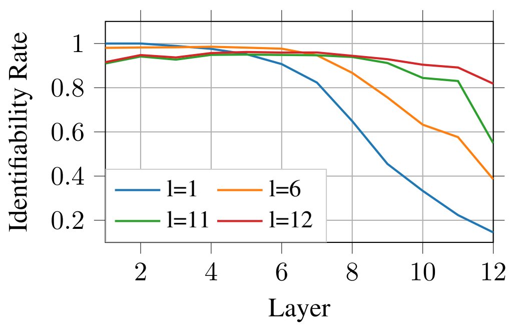
To investigate this assumption we introduce the concept of *token identifiability*, as the existence of a one-to-one mapping between contextual embeddings and their corresponding input tokens. Formally, a token $\boldsymbol{e}_i^l$ is identifiable if there exists a function $g(\cdot)$ such that $g(\boldsymbol{e}_i^l) = \boldsymbol{x}_i$. We cannot prove the existence of g analytically. Instead, for each layer l we use a function approximator $\hat{g}_l(\boldsymbol{e}_i^l) = \hat{\boldsymbol{x}}_i$ trained on a dataset of $(\boldsymbol{e}_i^l, \boldsymbol{x}_i)$ pairs. We then search for the nearest neighbour $\hat{\boldsymbol{x}}_i^{1nn}$ of $\hat{\boldsymbol{x}}_i$ within the same sentence, and say that $\boldsymbol{e}_j^l$ is identifiable if $\hat{\boldsymbol{x}}_i^{1nn} = \boldsymbol{x}_i$. For evaluation we then report the *token identifiability rate* defined as the percentage of correctly identified tokens.

4.1 Setup

For the experiments in this and subsequent sections we use the evaluation data set from the Microsoft Research Paraphrase Corpus (MRPC) dataset (Dolan & Brockett, 2005). The evaluation set contains 408 examples with a sequence length d_s between 26 and 92 tokens, with 58 tokens on average. We pass all 408 sentences (21,723 tokens) through BERT and extract for each token the input



(a)



(b)

Figure 2: (a) Identifiability of contextual word embeddings at different layers. Here, $\hat{g}$ is trained and tested on the same layer. We do 10-fold cross-validation: Solid lines show the mean, shaded areas show the min/max across all folds. (b) MLP trained on layer l and tested on all layers.

embeddings $\boldsymbol{x}_i$ and the hidden embeddings $\boldsymbol{e}_i^l$ at all layers. We then train $\hat{g}$ on the regression task of predicting input tokens $\boldsymbol{x}_i$ from hidden tokens $\boldsymbol{e}_i^l$. We use 70/15/15 train/validation/test splits and ensure that tokens from the same sentence are not split across sets. The validation set is used for early stopping. See Appendix B.1 for more details on the setup and training procedure.

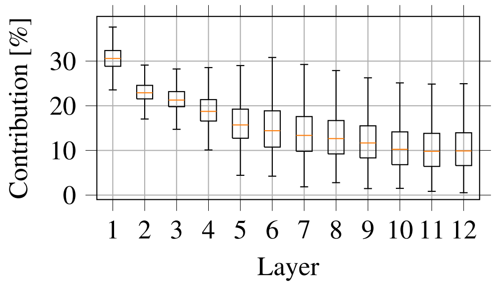
4.2 Experimental Results and Discussion

In a first experiment, we use a linear perceptron $\hat{g}_l^{lin}$ and a non-linear MLP $\hat{g}_l^{MLP}$, where training, validation and test data all come from layer l. Figure 2a shows the test set token identifiability rate of $\hat{g}_l$ for $l = [1, ..., 12]$. We also report a naive baseline $\hat{g}_l^{naive}(\boldsymbol{e}_i^l) = \boldsymbol{e}_i^l$, i.e., we directly retrieve the nearest neighbour of $\boldsymbol{e}_i^l$ from the input tokens. The results for $\hat{g}_l^{naive}$ show that contextual embeddings stay close to their input embeddings up to layer 4, followed by a linear decrease in token identifiability rate. $\hat{g}_l^{lin}$ and $\hat{g}_l^{MLP}$ are close, but perform considerably better than $\hat{g}_l^{naive}$. $\hat{g}_l^{MLP}$ starts outperforming $\hat{g}_l^{lin}$ after layer 6. In the last layer, $\hat{g}_l^{MLP}$ can recover 82% of tokens, and $\hat{g}_l^{lin}$ 73%. See Appendix B.2 for a comparison of train and test performance.

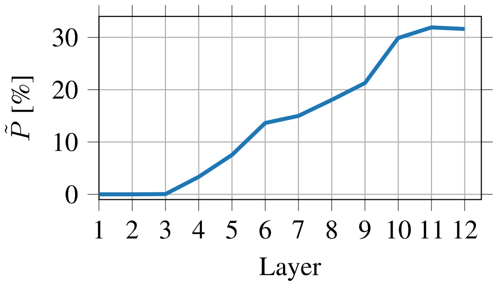
This experiment shows that tokens remain identifiable in the first layers of BERT. Starting from layer 3, the identifiability rate starts to decrease and at layer 12, almost 20% of hidden tokens cannot be mapped back to their input token anymore by $\hat{g}_l^{MLP}$. This suggests that the implicit assumption of token identifiability does not hold in later layers. Thus, one should confirm identifiability on a per-token level before interpreting contextual embeddings. $\hat{g}_l^{MLP}$ achieves roughly a 30% reduction in error at the last layer compared to $\hat{g}_l^{lin}$, indicating that the non-linearities in BERT play an important role in maintaining token identifiability. Finally, Lin et al. (2019) show that BERT discards much of the positional information after layer 3. This is also reflected in our results by the decrease in identifiability rate after layer 3. However, tokens remain largely identifiable throughout the model, indicating that BERT does not only rely on the positional embeddings to track token identity.

In a second experiment we test how well the $\hat{g}_l^{MLP}$ trained only on $(\boldsymbol{e}_i^l, \boldsymbol{x}_i)$ pairs from one layer l generalizes to all layers, see Figure 2b. For $l = 1$, the token identifiability rate on subsequent layers drops quickly to 91% at layer 6 and 15% at layer 12. Interestingly, for $l = 12$ a very different pattern can be observed, where the identifiability is 82% for layer 12 and then increases when testing on earlier layers. Further, for $l = 6$ we see both patterns.

This experiment suggests that the nature of token identity changes as tokens pass through the model, and patterns learned on data from later layers transfer well to earlier layers. The experiment also shows that layer 12 is behaving differently than the other layers. In particular, generalizing *to* layer 12 from layer 11 seems to be difficult, signified by a sudden drop in token identifiability rate. We believe this is due to a task dependent parameter adaptation induced in the last layer by the next-sentence prediction task which only uses the CLS token. See Appendix B.3 for results of $\hat{g}_l^{lin}$ and $\hat{g}_l^{MLP}$ for all layers.



(a)



(b)

Figure 3: (a) Contribution of the input token to the embedding at the same position. The orange line represents the median value and outliers are not shown. (b) Percentage of tokens $\tilde{P}$ that are *not* the main contributors to their corresponding contextual embedding at each layer.

5 Attribution Analysis to Identify Context Contribution

We conclude by considering the role of the contextual information in the hidden embeddings. We introduce *Hidden Token Attribution*, a context quantification method based on gradient attribution (Simonyan et al., 2014) to investigate the hidden tokens' sensitivity with respect to the input tokens.

5.1 Hidden Token Attribution

Gradient based attribution approximates the neural network function $f(X)$ around a given sequence of input word embeddings $X \in \mathbb{R}^{d_s \times d}$ by the linear part of the Taylor expansion. With this, the network sensitivity is analyzed by looking at how small changes at the input correlate with changes at the output. Since in the linear approximation this change is given by the gradient $\nabla_{x_i} f = \frac{\delta f(X)}{\delta \boldsymbol{x}_i}$ for a change in the i-th input token $\boldsymbol{x}_i \in \mathbb{R}^d$ of X, the attribution of how much input token $\boldsymbol{x}_i$ affects the network output $f(X)$ can be approximated by the L_2 norm of the gradient: $\text{attr}(x_i) = ||\nabla_{x_i} f||_2$. Since we are interested in how much a given hidden embedding $\boldsymbol{e}_j^l$ attributes to the input tokens $\boldsymbol{x}_i$, $i \in [1, 2, \ldots, d_s]$, we define the relative input contribution $c_{i,j}^l$ of input $\boldsymbol{x}_i$ to output $f(X) = \boldsymbol{e}_j^l$ as

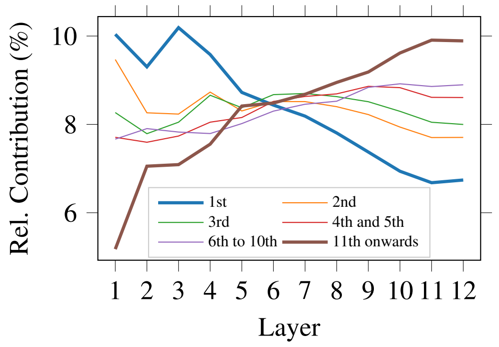
$$c_{i,j}^l = \frac{||\nabla_{i,j}^l||_2}{\sum_{k=0}^{d_s} ||\nabla_{k,j}^l||_2} \qquad \text{with} \quad \nabla_{i,j}^l = \frac{\delta \boldsymbol{e}_j^l}{\delta \boldsymbol{x}_i}$$

Since we normalize by dividing by the sum of the attribution values to all input tokens, we obtain values between 0 and 1 that represent the *contribution* of each input token x_i to the hidden embedding e_j^l. *Hidden Token Attribution* differs from the standard use of gradient attribution in that, instead of taking the gradients of the output of the model with respect to the inputs in order to explain the model's decision, we calculate the contribution of the inputs to intermediate embeddings in order to track the mixing of information. Further details of this method are discussed in Appendix C.1.

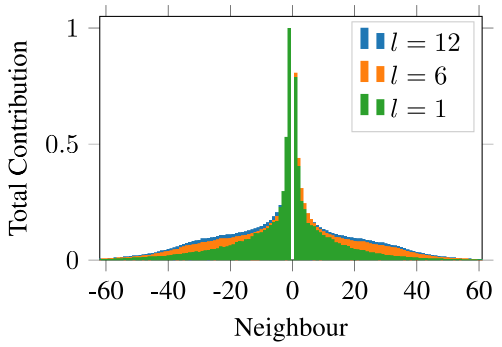
5.2 Token Mixing: Contribution of Input Tokens

We use Hidden Token Attribution to extend the token identity results of Section 4 by showing *how much* of the input token is actually contained in a given hidden embedding. In Figure 3a we report the contribution $c_{j,j}^l$ of input tokens $\boldsymbol{x}_j$ to their corresponding hidden embeddings $\boldsymbol{e}_j^l$ at the same position j for each layer l. Already after the first layer the median contribution of the input token is less than a third (30.6%). The contribution then decreases monotonically with depth; at layer 6 the median contribution is only 14.4% and after the last layer it is 10.7%. In Appendix C.5 we provide detailed results by word type.

Next, we study which input token is the largest contributor to a given hidden embedding $\boldsymbol{e}_j^l$. We observe that the corresponding input token $\boldsymbol{x}_j$ generally has the largest contribution. Figure 3b shows the percentage $\tilde{P}$ of tokens that are *not* the highest contributor to their hidden embedding at each layer. In the first three layers the original input $\boldsymbol{x}_j$ always contributes the most to the embedding



(a)



(b)

Figure 4: (a) Relative contribution per layer of neighbours at different positions. (b) Total contribution per neighbour for the first, middle and last layers.

$\boldsymbol{e}_j^l$. In subsequent layers, $\tilde{P}$ increases monotonically, reaching 18% in the sixth layer and 30% in the last two layers.

These results show that, starting from layer three, self-attention strongly mixes the input information by aggregating the context into the hidden embeddings. This is in line with the results from Section 4, where we see a decrease in token identifiability rate after layer three. However, $\tilde{P}$ is always higher than the token identifiability error at the same layer, indicating that tokens are mixed in a way that often permits recovering token identity even if the contribution of the original token is outweighed by other tokens. This implies that there is some "identity information" that is preserved through the layers. The strong mixing of information further questions the common assumption that attention distributions can be interpreted as "how much a word attends to another word".

5.3 Contribution of Context to Hidden Tokens

In this section we study how context is aggregated into hidden embeddings. Figure 4a shows the relative contribution of neighbouring tokens at each layer for the relative positions: first, second, third, fourth and fifth together, sixth to 10th together, and the rest. The closest neighbours (1st) contribute significantly more in the first layers than in later layers. Conversely, the most distant neighbours (11th onwards) contribute the most in deeper layers (cf. Appendix C.2).

Despite the progressive increase in long-range dependencies, the context in the hidden embeddings remains mostly local. Figure 4b represents the normalized total contribution aggregated over all tokens from each of their neighbours at the first, middle and last layer. This figure shows that the closest neighbours consistently contribute the most to the contextual word embedding regardless of depth. On the other hand, we indeed observe an increase of distant contributions at later layers.

The results of this section imply that transformers learn local operators from data in an unsupervised manner, in the absence of any such prior in the architecture. This behavior is not obvious, since attention is a highly non-local operator, and in turn indicates the importance of local dependencies in natural language. While contribution is local *on average*, we find that there are exceptions, such as the [CLS] token (cf. Appendix C.3). Furthermore, using our Hidden Token Attribution method, one can track how context is aggregated for specific tokens (cf. Appendix C.4).

6 Related Work

Input-output mappings play an important role in NLP. In machine translation, for example, they were originally introduced in the form of explicit *alignments* between source and target words (Brown et al., 1993). Neural translation architectures re-introduced this concept early, in the form of *attention* (Bahdanau et al., 2015). The development of multi-head self-attention (Vaswani et al., 2017) has led to a wide range of impressive results in NLP. Encouraged by the success of attention based models, much work has been devoted to gaining a better understanding of what these models learn. A particular focus has been placed on using attention distributions to explain model decisions.

Jain & Wallace (2019) show that attention distributions of LSTM based encoder-decoder models are not unique, and that adversarial attention distributions that do not change much the model's output can be constructed. They further show that attention distributions only correlate weakly to moderately with dot-product based gradient attribution. Wiegreffe & Pinter (2019) also find that adversarial attention distributions can be easily found, but that these alternative distributions perform worse on a simple diagnostic task. Serrano & Smith (2019) find that zero-ing out attention weights based on gradient attribution changes the output of a multi-class prediction task more quickly than zero-ing out based on attention weights, thus showing that attention is not the best predictor of learned feature importance. These papers differ in their approaches, but they all provide empirical evidence showing that attention distributions are not unique with respect to downstream parts of the model (e.g., output) and hence should be interpreted with care. Here, we support these empirical findings by presenting a theoretical proof of the identifiability of attention weights: when the sequence length is larger than the attention head dimension, the attention weights are not identifiable due to the non-trivial null space. Further, while these works focus on RNN-based language models with a single layer of attention, we instead consider multi-head multi-layer self-attention models. Our token classification and token mixing experiments show that the percentage of non-identifiable tokens increases with depth. These results further reinforce the point that the factors that contribute to the mixing of information are complex and deserve further study.

Voita et al. (2019) and Michel et al. (2019) find that only a small number of heads in BERT seem to have a relevant effect on the output. These results are akin to our conclusions about the non-identifiability of the attention weights, showing that a significant part of the attention weights do not affect downstream parts of the model. One specific line of work investigates the internal representations of transformers by attaching probing classifiers to different parts of the model. Tenney et al. (2019) find that BERT has learned to perform steps from the classical NLP pipeline. Similarly, Jawahar et al. (2019) show that lower layers of BERT learn syntactic features, while higher layers learn semantic features. They also argue that long-range features are learned in later layers, which agrees with our attribution-based experiments.

7 Conclusion

In this work we explore the concept of identifiability in transformers from different yet complementary angles. We first prove that attention weights are non-identifiable when the sequence length is longer than the attention head dimension, implying that infinitely many attention distributions can lead to the same internal representation and model output. We therefore propose *effective attention* as a tool to improve the interpretability of attention weights. Second, we show that non-linearities in transformers help maintaining the identifiability of hidden embeddings. However, input tokens gradually lose their identity after the early layers. Finally, we present Hidden Token Attribution, a gradient-based method to quantify information mixing, and we use it to demonstrate that input tokens get heavily mixed inside transformers. Therefore, attention-based attribution, which suggests that a word at some layer is attending to a specific input word, can be misleading. Rather, a hidden mixture of embeddings is attending to other hidden mixtures of embeddings. We further show that context is progressively aggregated into the hidden embeddings while some identity information is preserved. Moreover, we show that context aggregation is mostly local and that distant dependencies become relevant only in the last layers, which highlights the importance of local information for natural language understanding.

In summary, we show that representations learned via self-attention are the result of complex factorizations that can be further characterized. We hope our work is a step towards a deeper understanding of this process. Our results suggest that some of the conclusions in prior work (Vaswani et al., 2017; Vig, 2019; Marecek & Rosa, 2018; Clark et al., 2019; Raganato & Tiedemann, 2018; Voita et al., 2019; Tang et al., 2018; Wangperawong, 2018; Padigela et al., 2019; Baan et al., 2019; Lin et al., 2019; Dehghani et al., 2019; Zenkel et al., 2019; Klein & Nabi, 2019; Coenen et al., 2019) may be worth re-examining using the tools introduced here.

REFERENCES

Chris Alberti, Kenton Lee, and Michael Collins. A BERT Baseline for the Natural Questions. *https://arxiv.org/abs/1901.08634*, 2019.

Joris Baan, Maartje ter Hoeve, Marlies van der Wees, Anne Schuth, and Maarten de Rijke. Do transformer attention heads provide transparency in abstractive summarization? *CoRR*, abs/1907.00570, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

R. Bellman and Karl Johan Åström. On structural identifiability. *Mathematical Biosciences*, 7: 329–339, 1970.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19 (2):263–311, 1993. URL `https://www.aclweb.org/anthology/J93-2003`.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341, 2019.

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda B. Viégas, and Martin Wattenberg. Visualizing and measuring the geometry of BERT. *CoRR*, abs/1906.02715, 2019.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*, 2005.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pp. 249–256, 2010. URL `http://proceedings.mlr.press/v9/glorot10a.html`.

Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. URL `http://arxiv.org/abs/1606.08415`.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 3543–3556, 2019.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 3651–3657, 2019. URL `https://www.aclweb.org/anthology/P19-1356/`.

Marcin Junczys-Dowmunt. Microsoft translator at wmt 2019: Towards large-scale document-level neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pp. 225–233, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Tassilo Klein and Moin Nabi. Attention is (not) all you need for commonsense reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 4831–4836, 2019.

Yongjie Lin, Yi Chern Tan, and Robert Frank. Open sesame: Getting inside bert's linguistic knowledge. *arXiv preprint arXiv:1906.01698*, 2019.

David Marecek and Rudolf Rosa. Extracting syntactic trees from transformer encoder self-attentions. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pp. 347–349, 2018.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *CoRR*, abs/1905.10650, 2019. URL `http://arxiv.org/abs/1905.10650`.

Harshith Padigela, Hamed Zamani, and W. Bruce Croft. Investigating the successes and failures of BERT for passage re-ranking. *CoRR*, abs/1905.01758, 2019.

Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 1499–1509, 2018.

Nina Pörner, Hinrich Schütze, and Benjamin Roth. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 340–350, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Alessandro Raganato and Jörg Tiedemann. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pp. 287–297, 2018.

Sofia Serrano and Noah A. Smith. Is attention interpretable? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 2931–2951, 2019. URL `https://www.aclweb.org/anthology/P19-1282/`.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014.

Gongbo Tang, Rico Sennrich, and Joakim Nivre. An analysis of attention mechanisms: The case of word sense disambiguation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pp. 26–35, 2018.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 4593–4601, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017.

Jesse Vig. Visualizing attention in transformer-based language representation models. *CoRR*, abs/1904.02679, 2019.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 5797–5808, 2019.

Artit Wangperawong. Attending to mathematical language with transformers. *CoRR*, abs/1812.02825, 2018.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *CoRR*, abs/1908.04626, 2019. URL `http://arxiv.org/abs/1908.04626`.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. Modeling localness for self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 4449–4458, 2018. URL `https://aclanthology.info/papers/D18-1475/d18-1475`.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *https://arxiv.org/abs/1906.08237*, 2019.

Thomas Zenkel, Joern Wuebker, and John DeNero. Adding interpretable attention to neural translation models improves word alignment. *CoRR*, abs/1901.11359, 2019.

A Identifiability of Self-Attention

A.1 Background on attention identifiability

Often, the identifiability issue arises for a model with a large number of unknown parameters and limited observations. Taking a simple linear model $y = x_1\beta_1 + x_2\beta_2$ as an example, when there is only one observation (y, x_1, x_2), model parameters β_1 and β_2 cannot be uniquely determined. Moreover, in the matrix form $Y = X\beta$, by definition the parameter β is identifiable only if $Y = X\beta_1$ and $Y = X\beta_2$ imply $\beta_1 = \beta_2$. So if the null space contains only the zero solution $\{\beta | X\beta = 0\} = \{0\}$, i.e., $X\beta_1 - X\beta_2 = X(\beta_1 - \beta_2) = 0 \implies \beta_1 - \beta_2 = 0$, the model is identifiable. Therefore, the identifiability of parameters in a linear model is linked to the dimension of the null space, which in the end is determined by the rank of X.

A.2 Additional Results of the Effective Attention vs. Raw Attention Results

In Figure 5 we provide a recreation of the Figure regarding the attention of tokens towards the [SEP] token found in (Clark et al., 2019) with average attention as well as average effective attention. Again we see that most of the raw attention lies effectively in the Null space, rendering the conclusions drawn questionable. The Figures are produced using the code from Clark et al. (2019).

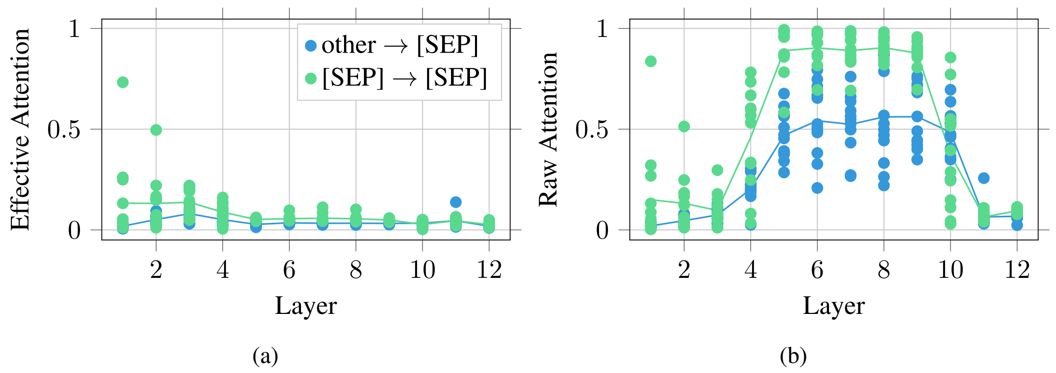


Figure 5: Effective attention (a) vs. raw attention (b). (a) Each point represents the average effective attention from a token type to a token type. Solid lines are the average effective attention of corresponding points in each layer. (b) is the corresponding figures using raw attention weights.

B Token Identifiability Experiments

B.1 Experimental Setup and Training Details

The linear perceptron and MLP are both trained by minimizing the L2 loss using the ADAM optimizer (Kingma & Ba, 2015) with a learning rate of $\alpha = 0.0001$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a batch size of 256. We monitor performance on the validation set and stop training if there is no improvement for 20 epochs. The input and output dimension of the models is $d = 768$; the dimension of the contextual word embeddings. For both models we performed a learning rate search over the values $\alpha \in [0.003, 0.001, 0.0003, 0.0001, 0.00003, 0.00001, 0.000003]$. The weights are initialized with the Glorot Uniform initializer (Glorot & Bengio, 2010). The MLP has one hidden layer with 1000 neurons and uses the gelu activation function (Hendrycks & Gimpel, 2016), follwing the feed-forward layers in BERT and GPT. We chose a hidden layer size of 1000 in order to avoid a bottleneck. We experimented with using a larger hidden layer of size 3072 and adding dropout to more closely match the feed-forward layers in BERT, which only resulted in increased training times and we hence deferred from further architecture search.

We split the data by sentences into train/validation/test according to a 70/15/15 split. This way of splitting the data ensures that the models have never seen the test sentences (i.e., contexts) during training. In order to get a more robust estimate of performance we perform the experiments in Figure 2a using 10-fold cross validation. The variance, due to the random assignment of sentences to train/validation/test sets, is small. We thus do not perform 10-fold cross validation for the experiments in Figure 2b, also because we are doing a relative comparison of the same model trained using the same train/validation/test splits.

B.2 Generalization Error

Figure 6 shows the token identifiability rate for train and test set for both models. Both models are overfitting to the same degree. The fact that the linear model has about the same generalization error as the MLP suggests that more training data would not significantly increase performance on the test set. Further, we trained the MLP on layer 11 using 50%, 80%, 90% and 100% of the training data set. The MLP achieved the following token identifiability rate on the test set: 0.74, 0.8, 0.81, 0.82. This indicates that the MLP would not profit much from more data.

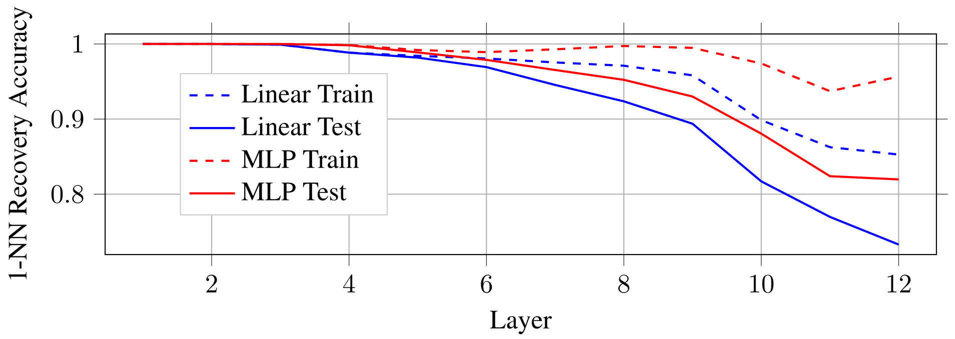


Figure 6: Train and test token identifiability rates for the linear perceptron and MLP.

B.3 Additional results for Figure 2b

For better readability, Figure 2b in the main text only shows results of the MLP trained on layers $l = [1, 6, 11, 12]$ and tested on all other layers. Figures 7 and 8 show the complete results for the MLP and linear perceptron trained on layer $l = [1, ..., 12]$ and tested on all layers. The linear and non-linear perceptrons show similar trends. However, the MLP trained on layer 10 behaves strangely. We verified that this result persists for different random seeds leading to different weight initializations and dataset splits. We hypothesize that the MLP overfits to a spurious pattern in layer 10 which does not generalize well to previous layers.

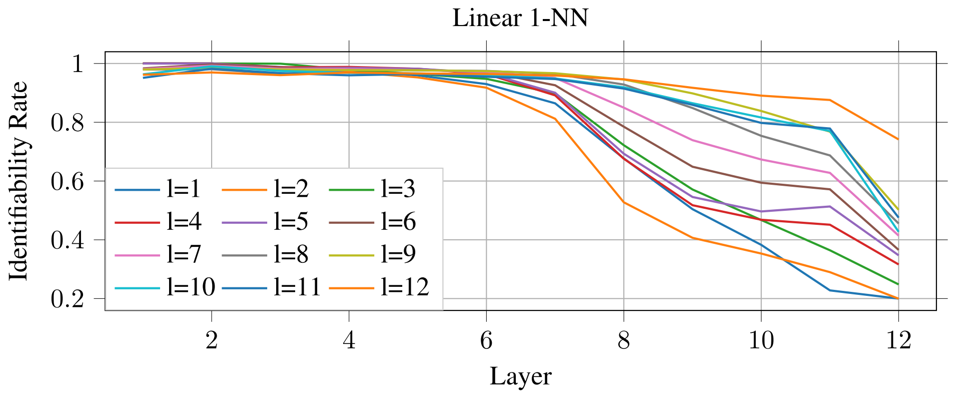


Figure 7: Linear Perceptron generalizing to all layers

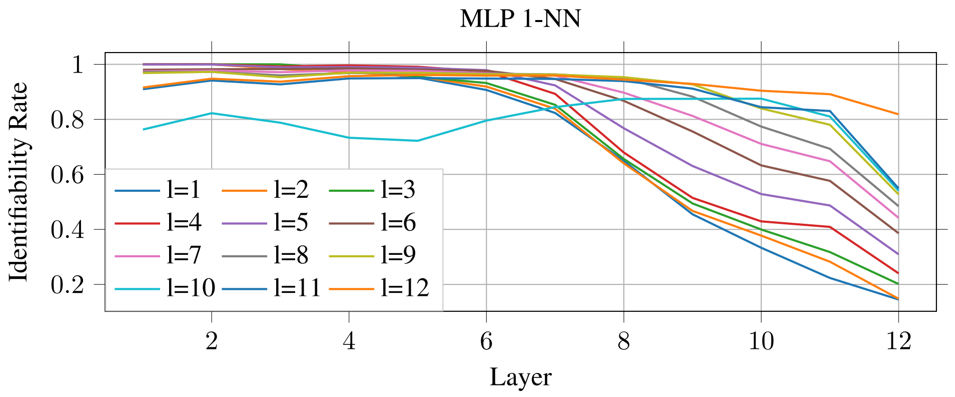


Figure 8: MLP generalizing to all layers

C Context Contribution Analysis

C.1 Hidden Token Attribution: Details

The attribution method proposed in Section 5.1 to calculate the contribution of input tokens to a given embedding does not look at the output of the model but at the intermediate hidden representations and therefore is task independent. Since the contribution values do not depend on the task that is evaluated, we can compare these values directly to attention distributions, which are also task-independent. In this way, we can compare to other works in the literature (Vig, 2019; Clark et al., 2019; Klein & Nabi, 2019; Coenen et al., 2019; Lin et al., 2019) by using the publicly available pretrained BERT model in our analyses without fine-tuning it to a specific task.

Furthermore, since we are not interested in analysing how the input affects the output of the model but in quantifying the absolute contribution of the input tokens to the hidden embeddings, we use the L_2 norm of the gradients. If we were analyzing whether the input contributed positively or negatively to a given decision, the dot-product of the input token embedding with the gradient would be the natural attribution choice (Pörner et al., 2018).

C.2 Context Identifiability: Details

To calculate the relative contribution values shown in Figure 4a we firstly calculate the mean of the left and right neighbours for each of the groups of neighbours, i.e., first, second, third, fourth and fifth, sixth to 10th and, from 11th onwards. Then we aggregate the values averaging over all the tokens in the MRPC evaluation set. Finally, we normalize for each group so that the sum of the contribution values of each group is one. In this way, we can observe in which layer the contribution of a given group of neighbours is the largest.

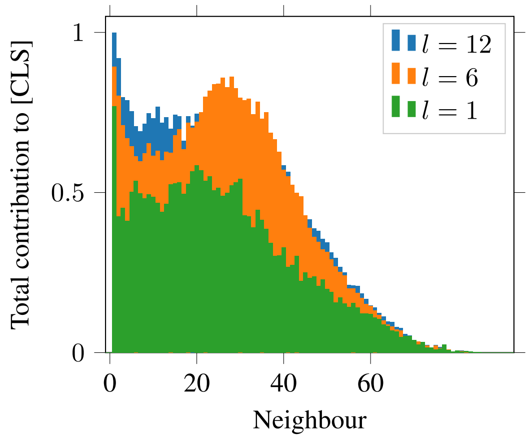
Our results on context identifiability from Section 5.3 complement some of the studies in previous literature. In (Jawahar et al., 2019) the authors observe that transformers learn local syntactic tasks in the first layers and long range semantic tasks in the last layers. We explain this behavior from the point of view of context aggregation by showing that distant context acquires more importance in the last layers (semantic tasks) while the first layers aggregate local context (syntactic tasks). Furthermore, the results showing that the context aggregation is mainly local, specially in the first layers, provide an explanation for the increase in performance observed in (Yang et al., 2018). In that work, the authors enforce a locality constraint in the first layers of transformers, which pushes the model towards the local operators that it naturally tends to learn, as we show in Figure 4b, improving in this way the overall performance.

C.3 Context Contribution to CLS token

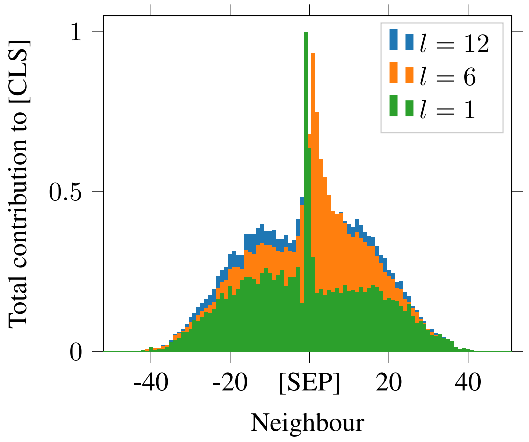
In this section we use Hidden Token Attribution to look at the contribution of the context to the [CLS] token, which is added to the beginning of the input sequence by the BERT pre-processing pipeline. This is an especially interesting token to look at because the decision of BERT for a classification task is based on the output in the [CLS] token. Furthermore, like the [SEP] token, it does not correspond to a natural language word and its position in the input sequence does not have any meaning. Therefore, the conclusion that context is on average predominantly local (cf. Section 5.3), is likely to not hold for [CLS].

The second and final pre-training task that BERT is trained on is next sentence prediction. During this task, BERT receives two sentences separated by a [SEP] token as input, and then has to predict whether the second sentence follows the first sentence or not. Therefore, it is expected that the context aggregated into the [CLS] token comes mainly from the tokens around the first [SEP] token, which marks the border between the first and second sentence in the input sequence. In Figure 9 we show the contribution to the [CLS] token from all of its neighbours averaged over all the examples in the MRPC evaluation set for the first, middle and last layers. In Figure 9a, the [CLS] token is placed at position 0 and we see how the context contribution comes mainly from the tokens around position 30, which is roughly the middle of the input examples. In Figure 9b we center the contribution around the first [SEP] token and indeed, it becomes clear that the [CLS] token is aggregating most of its context from the tokens around [SEP], i.e., from the junction between both sentences. In particular, the two tokens with the highest contribution are the tokens directly before and after [SEP]. Also, it seems that the second sentence contributes more to [CLS] than the first one.

These results give an insight on what information BERT uses to solve next sentence prediction and serves as an illustrative example of how Hidden Token Attribution can be used to analyze transformers.



(a)



(b)

Figure 9: Normalized total contribution to the [CLS] token (a) centered around [CLS] at position 0 (b) centered around [SEP].

C.4 TRACKING CONTEXT CONTRIBUTION

Here we show examples of how Hidden Token Attribution can track how context is aggregated for a given word at each layer. For reasons of space we show only few words of a randomly picked sentence of the MRPC evaluation set, which is tokenized as follows:

```
[CLS] he said the foods ##er ##vic ##e pie business doesn ’ t fit
the company ’ s long - term growth strategy .  [SEP] " the foods
##er ##vic ##e pie business does not fit our long - term growth
strategy .  [SEP]
```

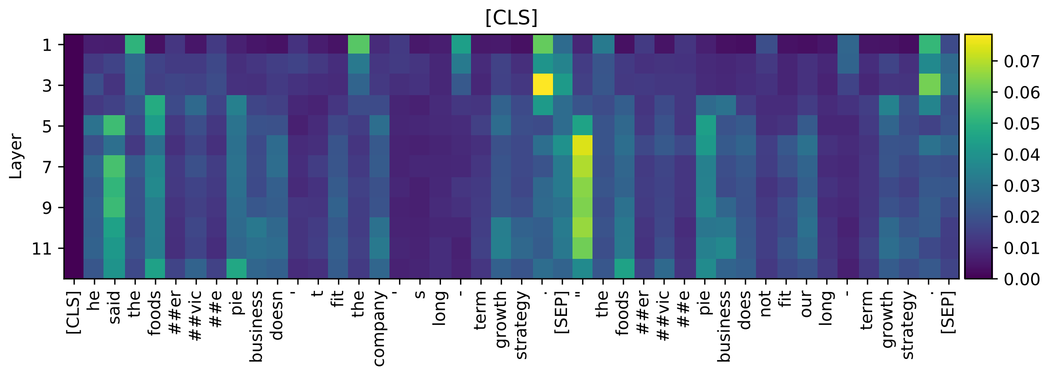


Figure 10: [CLS]

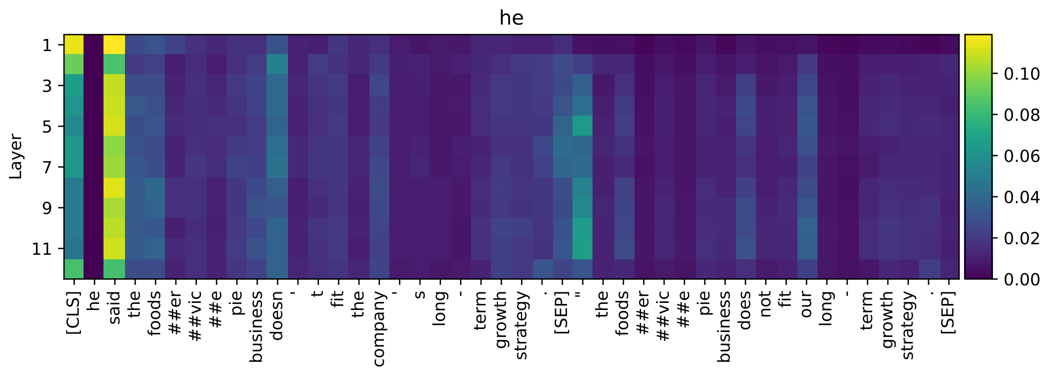


Figure 11: he

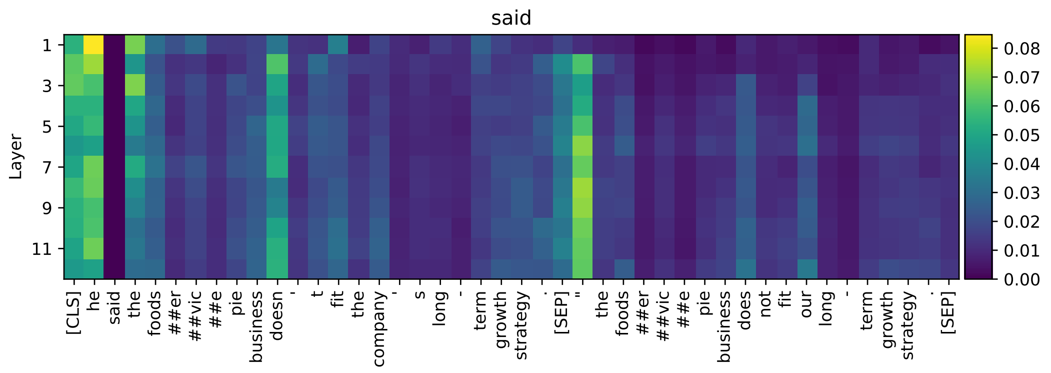


Figure 12: said

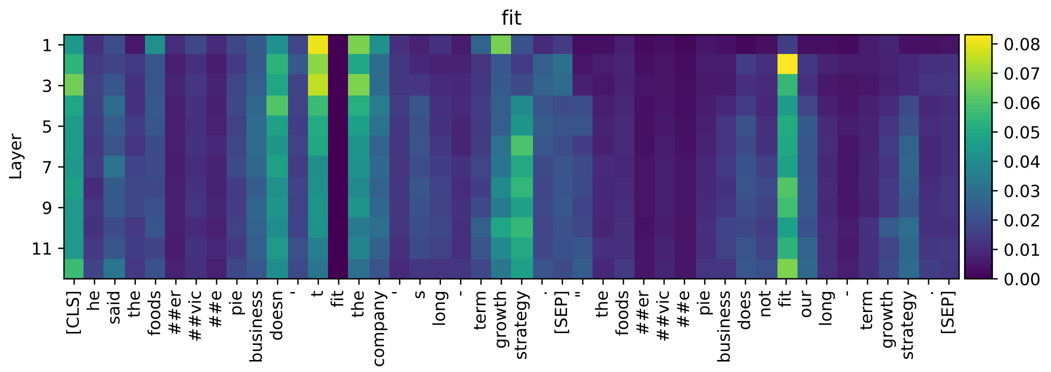


Figure 13: fit

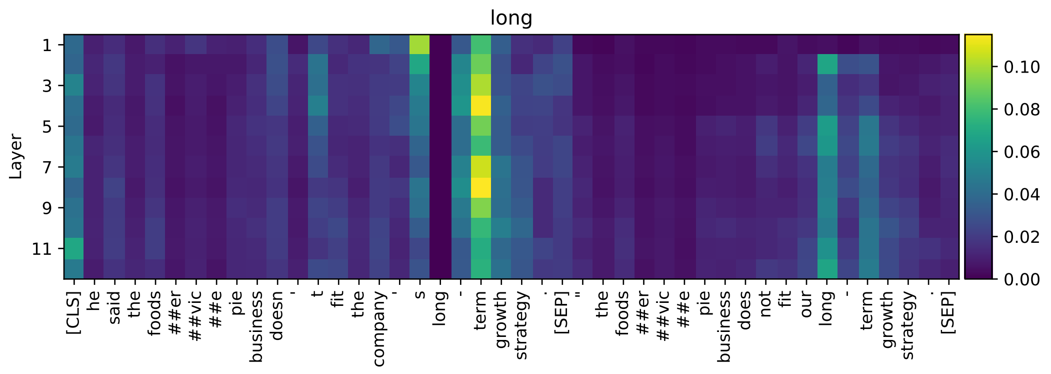


Figure 14: said

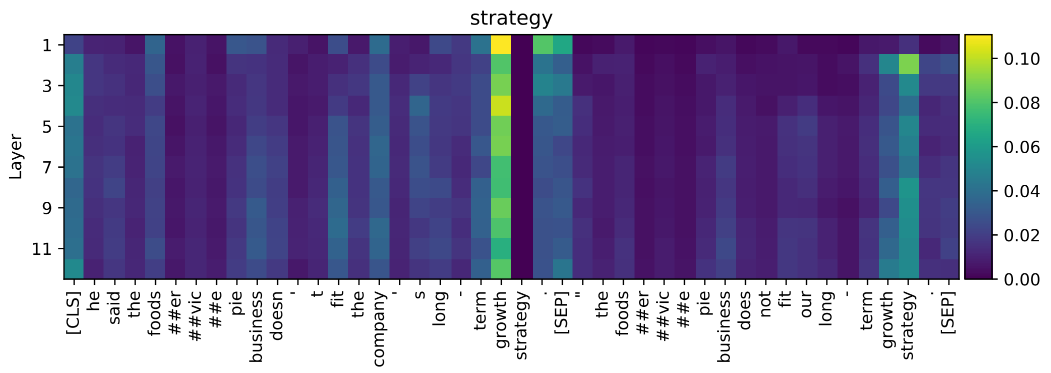


Figure 15: strategy

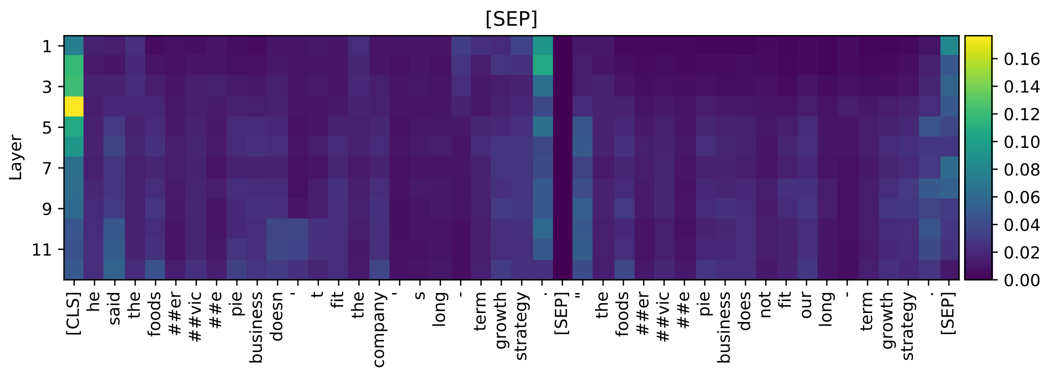


Figure 16: [SEP]

C.5 Token Contributions by POS Tag

Here we show the contribution of input tokens to hidden representations in all layers splitted by part-of-speech (POS) tag. The POS tags are ordered according to the contribution in layer 12.

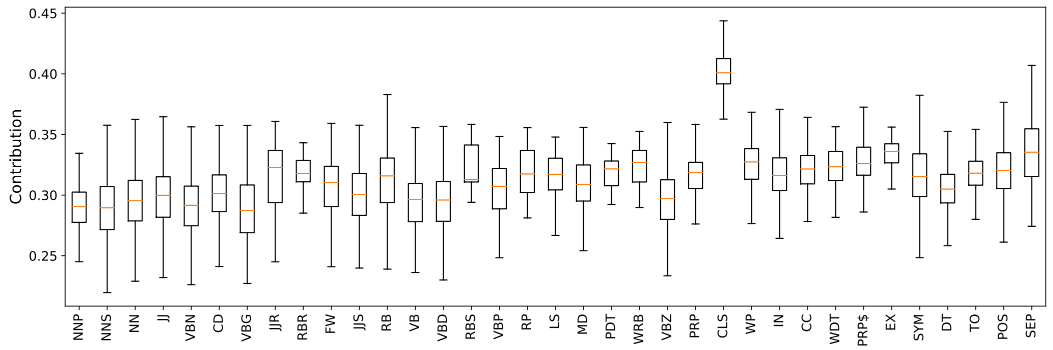


Figure 17: Layer 1

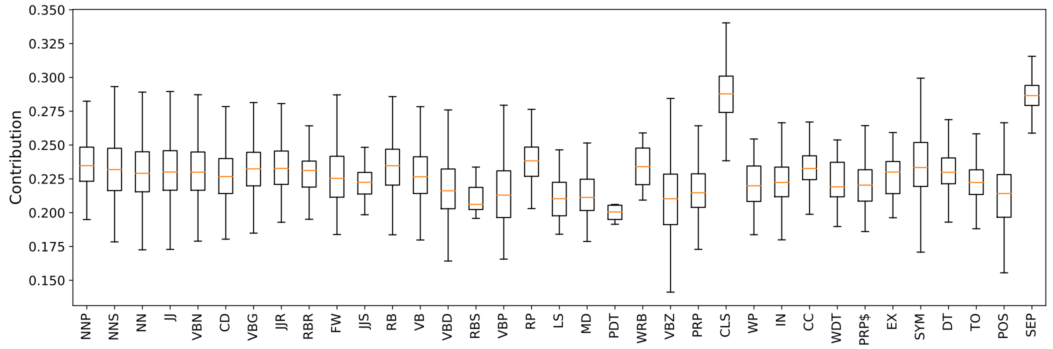


Figure 18: Layer 2

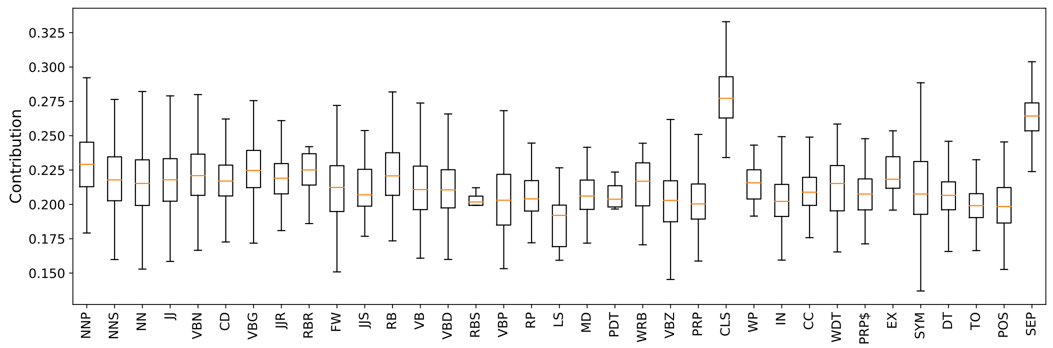


Figure 19: Layer 3

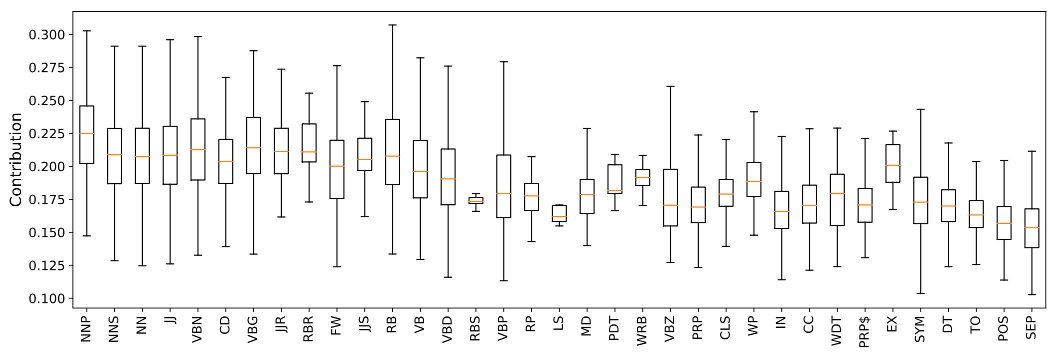


Figure 20: Layer 4

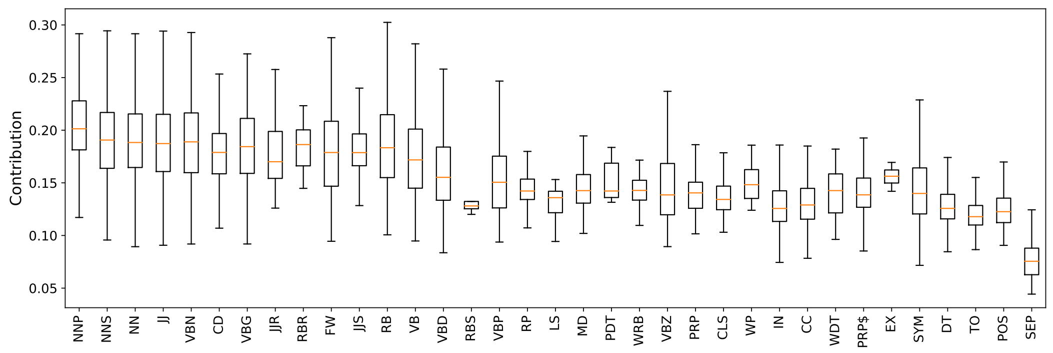


Figure 21: Layer 5

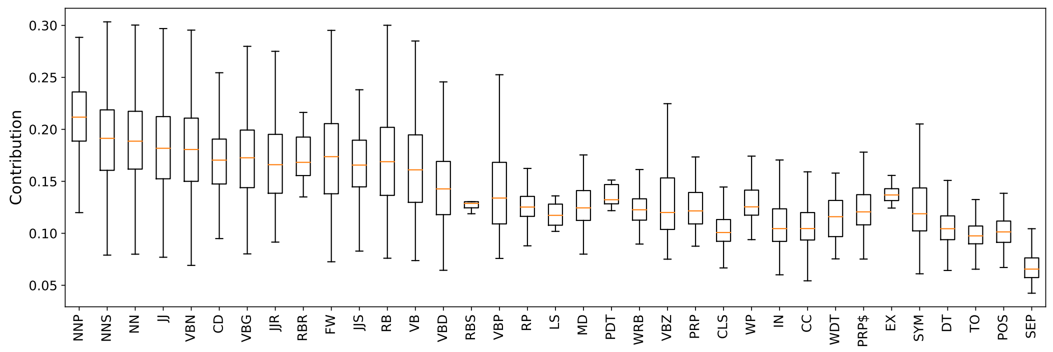


Figure 22: Layer 6

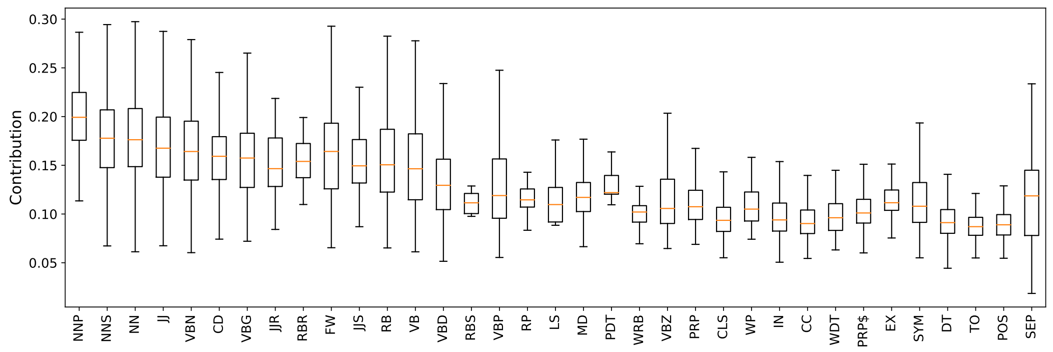


Figure 23: Layer 7

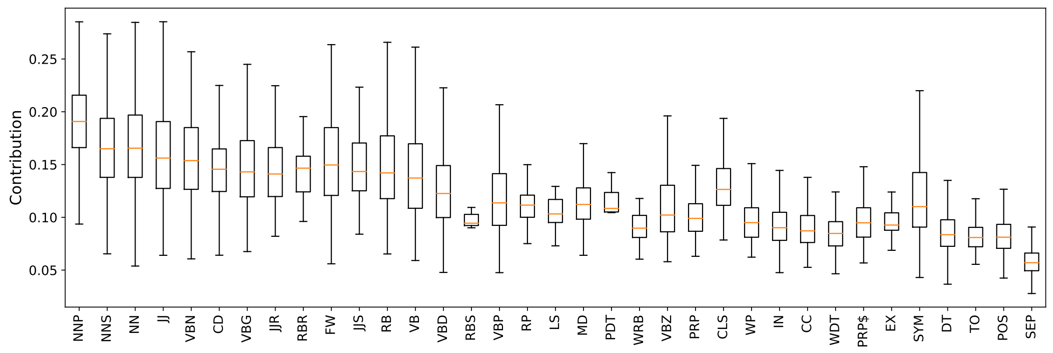


Figure 24: Layer 8

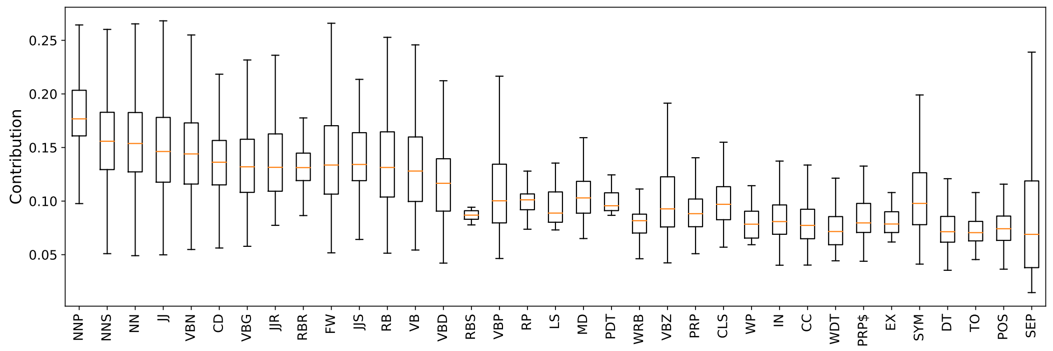


Figure 25: Layer 9

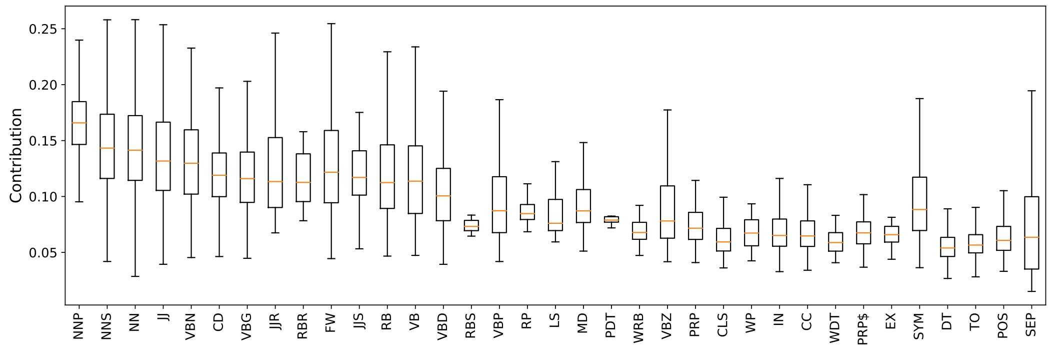


Figure 26: Layer 10

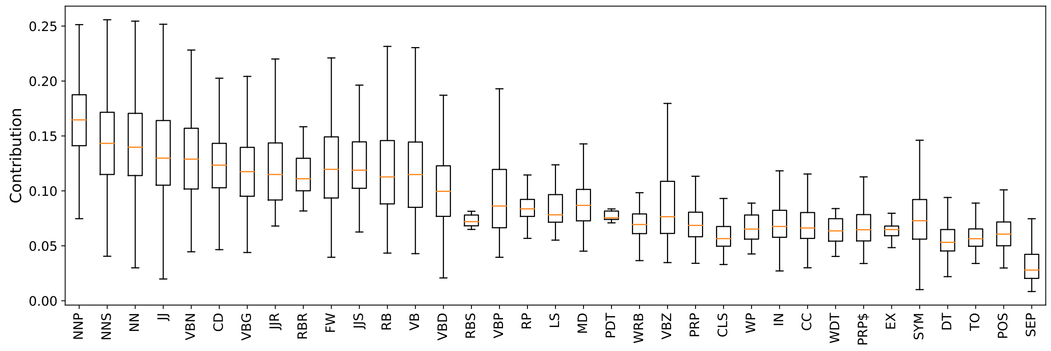


Figure 27: Layer 11

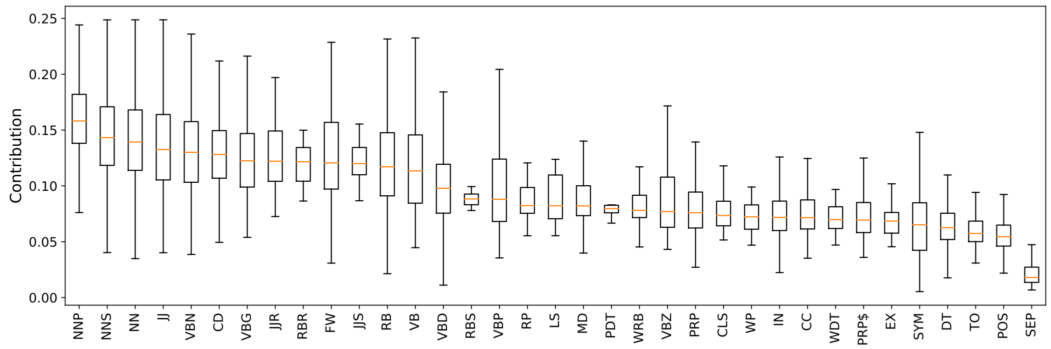


Figure 28: Layer 12