How do Decisions Emerge across Layers in Neural Models? Interpretation with Differentiable Masking

Nicola De Cao [1,2], Michael Schlichtkrull [1,2], Wilker Aziz [1], Ivan Titov [1,2]
[1]University of Amsterdam, [2]University of Edinburgh
{ nicola.decao, m.s.schlichtkrull, w.aziz } @uva.nl
ititov@inf.ed.ac.uk

Abstract

Attribution methods assess the contribution of inputs to the model prediction. One way to do so is *erasure*: a subset of inputs is considered irrelevant if it can be removed without affecting the prediction. Though conceptually simple, erasure's objective is intractable and approximate search remains expensive with modern deep NLP models. Erasure is also susceptible to the *hindsight bias*: the fact that an input can be dropped does not mean that the model 'knows' it can be dropped. The resulting pruning is over-aggressive and does not reflect how the model arrives at the prediction. To deal with these challenges, we introduce Differentiable Masking. DiffMask learns to mask-out subsets of the input while maintaining differentiability. The decision to include or disregard an input token is made with a simple model based on intermediate hidden layers of the analyzed model. First, this makes the approach efficient because we predict rather than search. Second, as with probing classifiers, this reveals what the network 'knows' at the corresponding layers. This lets us not only plot attribution heatmaps but also analyze how decisions are formed across network layers. We use DiffMask to study BERT models on sentiment classification and question answering.[1]

1 Introduction

Deep neural networks have become standard tools in NLP demonstrating impressive improvements over traditional approaches on many tasks (Goldberg, 2017). Their power typically comes at the expense of interpretability, which may prevent users from trusting predictions (Kim, 2015; Ribeiro et al., 2016), makes it hard to detect model or data deficiencies (Gururangan et al., 2018; Kaushik and Lipton, 2018) or verify that a model is fair and does not exhibit harmful biases (Sun et al., 2019; Holstein et al., 2019).

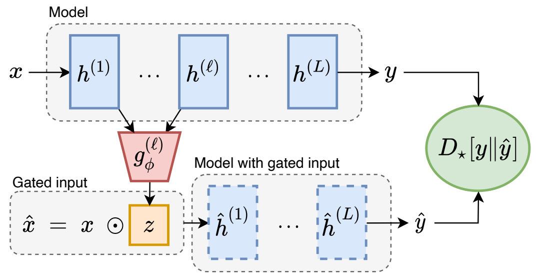
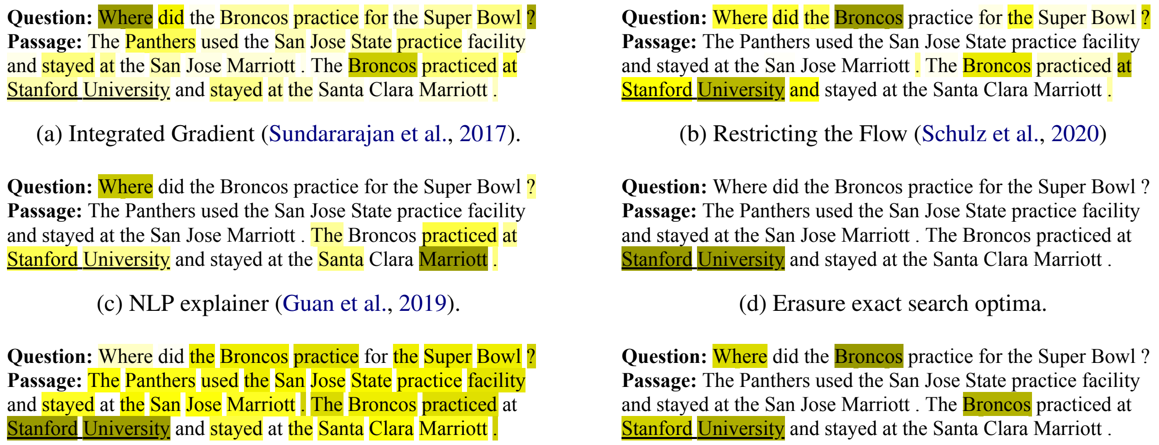


Figure 1: DiffMask: hidden states up to layer ℓ from a model (top) are fed to a classifier g that predicts a mask z. We use this to mask the input and re-compute the forward pass (bottom). The classifier g is trained to mask the input as much as possible without changing the output (minimizing a divergence $\mathrm{D}_\star$).

These challenges have motivated work on interpretability, both in NLP and generally in machine learning; see Belinkov and Glass (2019) and Jacovi and Goldberg (2020) for reviews. In this work, we study *post hoc interpretability* where the goal is to explain the prediction of a trained model and to reveal how the model arrives at the decision. This goal is usually approached with attribution methods (Bach et al., 2015; Shrikumar et al., 2017; Sundararajan et al., 2017), which explain the behavior of a model by assigning relevance to inputs.

One way to perform attribution is to use *erasure* where a subset of features (e.g., input tokens) is considered irrelevant if it can be removed without affecting the model prediction (Li et al., 2016; Feng et al., 2018). The advantage of erasure is that it is conceptually simple and optimizes a well-defined objective. This contrasts with most other attribution methods which rely on heuristic rules to define feature salience; for example, attention-based attri-

[1] Source code available at https://github.com/nicola-decao/diffmask



Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(a) Integrated Gradient (Sundararajan et al., 2017).

Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(b) Restricting the Flow (Schulz et al., 2020)

Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(c) NLP explainer (Guan et al., 2019).

Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(d) Erasure exact search optima.

Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(e) Our DIFFMASK.

Question: Where did the Broncos practice for the Super Bowl ?
Passage: The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott . The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott .

(f) Our DIFFMASK non-amortized.

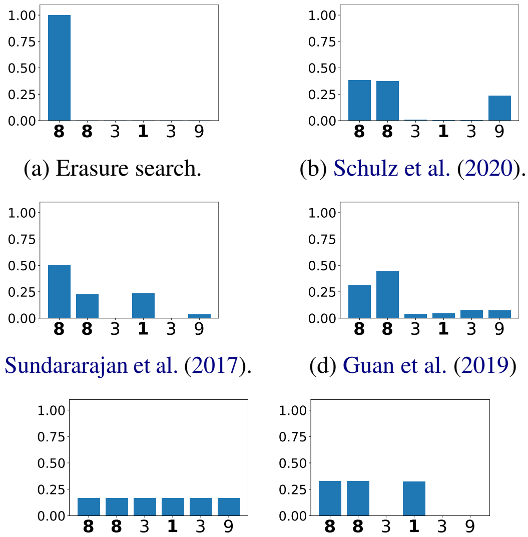
Figure 2: Question Answering token attribution: (b) and (c), are misleading (i.e., not faithful) as they attribute the prediction mostly to the answer span itself (underlined). Our method (d) reveals that the model pays attention to other named entities and the predicate ‘practice’ in both sentences. Predictions of the path-based methods (a) are more spread-out. Exact search (e) as well as approximate search (f) leads to pathological attributions.

bution (Rocktäschel et al., 2016; Serrano and Smith, 2019; Vashishth et al., 2019) or back-propagation methods (Bach et al., 2015; Shrikumar et al., 2017; Sundararajan et al., 2017). These approaches received much scrutiny in recent years (Nie et al., 2018; Sixt et al., 2020; Jain and Wallace, 2019), as they cannot guarantee that the network is ignoring low-scored features. They are often motivated as approximations of erasure (Baehrens et al., 2010; Simonyan et al., 2014; Feng et al., 2018) and sometimes evaluated using erasure as ground-truth (Serrano and Smith, 2019; Jain and Wallace, 2019).

Despite its conceptual simplicity, subset erasure is not commonly used in practice. First, it is generally **intractable**, and beam search (Feng et al., 2018) or leave-one-out estimates (Zintgraf et al., 2017) are typically used instead. These approximations may be inaccurate. For example, leave-one-out can underestimate the contribution of features due to saturation (Shrikumar et al., 2017). More importantly, even these approximations remain very expensive with modern deep (e.g., BERT-based; Devlin et al., 2019) models, as they require multiple computation passes through the model. Second, the method is **susceptible to the hindsight bias**: the fact that a feature *can be* dropped does not mean that the model ‘knows’ that it can be dropped and that the feature *is not* used by the model when processing the example. This results in over-aggressive pruning that does not reflect what information the model uses to arrive at the decision. The issue is pronounced in NLP tasks (see Figure 2d and Feng et al., 2018), though it is easier to see on an artificial example (Figure 3a). A model is asked to predict if there are more 8s than 1s in the sequence. The erasure attributes the prediction to a single 8 digit, as this reduced example yields the same decision as the original one. However, this does not reveal what the model was relying on: it has counted digits 8 and 1 as otherwise, it would not have achieved the perfect score on the test set.

We propose a new method, Differentiable Masking (DIFFMASK), which overcomes the aforementioned limitations and results in attributions that are more informative and help us understand how the model arrives at the prediction. DIFFMASK relies on learning sparse stochastic gates (a.k.a., masks), guaranteeing that the information from the masked-out inputs does not get propagated while maintaining end-to-end differentiability without having to resort to REINFORCE (Williams, 1992). The decision to include or disregard an input token is made with a simple model based on intermediate hidden layers of the analyzed model (see Figure 1). First, this *amortization* circumvents the need for combinatorial search making the approach efficient at test time. Second, as with probing classifiers (Adi et al., 2017; Belinkov and Glass, 2019), this reveals whether the network ‘knows’ at the corresponding layer what input tokens can be disregarded. During training inputs are *truly* masked whenever we sample zeros. After training, attribution scores correspond to the expectation of sampling non-zeros.

The amortization lets us not only plot attribution



(a) Erasure search.

(b) Schulz et al. (2020).

(c) Sundararajan et al. (2017).

(d) Guan et al. (2019)

(e) Our DiffMask conditioned on embedding layer (left) and hidden states (right).

Figure 3: Input attributions of several methods on a toy task: Given a sequence x of digits and a query $\langle n, m \rangle$ (8 and 1 in this example) of two digits, determine whether there are more n than m in x. Attributions are computed at the vector level and normalized to sum to 1.

heatmaps, as in Figure 2e, but also analyze how decisions are formed across network layers. In our artificial example, we see that in the bottom embedding layer the model cannot discard any tokens, as it does not 'know' which digits need to be counted (Figure 3e, left). In the second layer, it 'knows' that these are 8s and 1s, so the rest gets discarded (Figure 3e, right). In question answering (see Figure 8a), where we use a 24-layer model, it takes 13–16 layers for the model to 'realize' that *'Santa Clara Marriott'* is not relevant to the question and discard it. We also adapt our method to measuring the importance of intermediate states rather than inputs. This, as we discuss later, lets us analyze which states in every layer store information crucial for making predictions, giving us insights about the information flow.

Contributions We introduce DiffMask, a technique addressing limitations of attribution-based methods (especially erasure and its approximations), and demonstrate that it is stable and faithful to the analyzed models. We then use this technique to analyze BERT-based models fined-tuned on sentiment classification and question answering.

2 Method

We aim to understand how a trained model processes an input (i.e., a sequence of embedded tokens) to produce an output (e.g., a vector of class probabilities). First, for an input $x = \langle x_1, \ldots, x_n \rangle$, we obtain the output $y = f(x)$ of the model along with its hidden states $\langle h^{(0)}, \ldots, h^{(L)} \rangle$, where $h^{(0)} = x$. We then probe the model using a shallow *interpreter* network which takes hidden states up to a certain layer ℓ and outputs a binary mask $z = \langle z_1, \ldots, z_n \rangle$ indicating which input tokens are necessary and which can be disregarded. To assess whether the masked input $\hat{x} = \langle \hat{x}_1, \ldots, \hat{x}_n \rangle$ is sufficient, we re-feed the model with it and compute the output $\hat{y} = f(\hat{x})$. As long as $\hat{y}$ approximates the original output y well, we deem the inputs masked by z unnecessary.

Masking, however, as in multiplication by *zero*, makes a strong assumption about the geometry of the feature space, in particular, it assumes that the zero vector bears no information. Instead, we replace some of the inputs by a learned baseline vector b, i.e., $\hat{x}_i = z_i \cdot x_i + (1 - z_i) \cdot b$.

See Figure 1 for an overview. The interpreter model consists of $L+1$ classifiers, the ℓth of which conditions on the stack of hidden states up to $h^{(\ell)}$ to predict binary 'votes' $v^{(\ell)} = g_\phi^{(\ell)}(h^{(0)}, \ldots, h^{(\ell)})$ towards keeping or masking input tokens. Each classifier is a one-hidden-layer MLP, details and hyperparameters are provided in Appendix A. For a given depth ℓ, the interpreter decides to mask x_i out as soon as $v_i^{(k)} = 0$ for some $k \leq \ell$, i.e., $z_i = \prod_{k=0}^{\ell} v_i^{(k)}$. That is, in order to deem x_i unnecessary, it is sufficient to do so based on any subset of hidden states up until $h^{(\ell)}$.

Clearly, there is no direct supervision to estimate the parameters ϕ of the probe and the baseline b, thus we borrow erasure's objective: namely, we train the probe to mask-out as many input tokens as possible constrained to keeping $f(\hat{x}) \approx f(x)$. Since often, the output of f parameterizes a likelihood (e.g., a categorical distribution), we formulate the constraint in terms of a divergence $\mathrm{D}_\star$ between the two functions' outputs. We cast this, rather naturally, in the language of constrained optimization.

Objective A practical way to minimize the number of non-zeros predicted by g is minimizing the L_0 'norm'.[2] Thus, our $\mathcal{L}_0$ loss is defined as the

[2] L_0, denoted $\|z\|_0$ and defined as $\#(i|z_i \neq 0)$, is the number of non-zeros entries in a vector. Contrary to L_1 or L_2, L_0 is not a homogeneous function and, thus, not a proper norm. However, contemporary literature refers to it as a norm, and we do so as well to avoid confusion.

total number of positions that are not masked:

$$\mathcal{L}_0(\phi, b|x) = \sum_{i=1}^{n} \mathbf{1}_{[\mathbb{R}_{\neq 0}]}(z_i) \ , \tag{1}$$

where $\mathbf{1}(\cdot)$ is the indicator function. We minimize $\mathcal{L}_0$ for all data-points in the dataset $\mathcal{D}$ subject to a constraint that predictions from masked inputs have to be similar to the original model predictions:

$$\begin{aligned} \min_{\phi,b} \quad & \sum_{x\in\mathcal{D}} \mathcal{L}_0(\phi, b|x) \\ \text{s.t.} \quad & \mathrm{D}_\star[y\|\hat{y}] \leq m \quad \forall x \in \mathcal{D} \ , \end{aligned} \tag{2}$$

where $\hat{y} = f(\hat{x})$, $y = f(x)$, and the margin $m \in \mathbb{R}_{>0}$ is a hyperparameter. Since non-linear constrained optimisation is generally intractable, we employ Lagrangian relaxation (Boyd et al., 2004) optimizing instead

$$\max_{\lambda} \min_{\phi,b} \sum_{x\in\mathcal{D}} \mathcal{L}_0(\phi, b|x) + \lambda(\mathrm{D}_\star[y\|\hat{y}] - m) \ , \tag{3}$$

where $\lambda \in \mathbb{R}_{\geq 0}$ is the Lagrangian multiplier.

Stochastic masks Our objective poses two challenges: i) L_0 is discontinuous and has zero derivative almost everywhere, and ii) to output binary masks, g needs a discontinuous output activation such as the step function. A strategy to overcome both problems is to make the binary variables stochastic and treat the objective in expectation, in which case one option is to resort to REINFORCE (Williams, 1992), another is to use a sparse relaxation to binary variables (Louizos et al., 2018; Bastings et al., 2019). As we shall see (we compare the two aforementioned options in Table 2 and discuss them in Section 3.2), the latter proved more effective. Thus we opt to use the Hard Concrete distribution, a mixed discrete-continuous distribution on the closed interval $[0, 1]$. This distribution assigns a non-zero probability to exactly zero while it also admits continuous outcomes in the unit interval via the *reparameterization trick* (Kingma and Welling, 2014). We refer to Louizos et al. (2018) for details, but also provide a brief summary in Appendix B. With stochastic masks, the objective is computed in expectation, which addresses both sources of non-differentiability. Note that during training inputs are *truly* masked-out whenever we sample exact zeros. After training, attribution scores correspond to the expectation of sampling non-zero masks since any non-zero value corresponds to a leak of information.

Masking hidden states To reveal which hidden states store information necessary for realizing the prediction, we modify the probe slightly. For a given depth ℓ, we use a mask $z^{(\ell)} = g_\phi^{(\ell)}(h^{(\ell)})$ to replace some of the *states* in $h^{(\ell)} = \langle h_1^{(\ell)}, \ldots, h_n^{(\ell)} \rangle$ by a layer-specific baseline $b^{(\ell)}$, i.e. $\hat{h}_i^{(\ell)} = z_i^{(\ell)} \cdot h_i^{(\ell)} + (1 - z^{(\ell)}) \cdot b^{(\ell)}$. The resulting state $\hat{h}^{(\ell)}$ is used to re-compute subsequent states, $\hat{h}^{(\ell+1)}, \ldots, \hat{h}^{(L)}$, as well as the output, which we denote by $\hat{y}$. Here we do not aggregate 'votes' with a product because for this probe we want to discover whether hidden states are predictive of their own *usefulness*. See Figure 10 in Appendix D for an overview of this variant of DIFFMASK.

3 Experiments

The goal of this work is to uncover a *faithful* interpretation of an existing model, i.e. revealing, as accurately as possible, the process by which the model arrives at the prediction. Human-provided labels, such as human rationales (Camburu et al., 2018; DeYoung et al., 2020), will not help us in demonstrating this, as humans cannot judge if an interpretation is faithful (Jacovi and Goldberg, 2020). More precisely, human-provided labels do not show how the model behaves – e.g., annotations of what parts of the input are relevant for solving a particular task do *not* constitute a guarantee that a model relies on those parts more than others when making a prediction. When we evaluate an attribution method by comparing its outputs with human annotations, we are not measuring whether it provides faithful attributions but only if they are *plausible* according to humans. This goes against our goals as we aim to use the interpretation method to detect model deficiencies, which are usually cases where the model does not behave like humans. The ground-truth explanations of how a model makes certain predictions depend not only on the data but also on the model, and, unfortunately, are generally not known for real tasks and with complex models. This makes the evaluation and comparison of attribution methods non-trivial.

Our strategy is to i) show the effectiveness of DIFFMASK in a controlled setting (i.e., a toy task) where ground-truth is available; ii) test the effectiveness of our relaxation for learning discrete masks (on a real model for sentiment classification); and iii) demonstrate that the method is stable and models behave the same when masking is ap-

Methods	$\mathrm{D_{KL}} \downarrow$	$\mathrm{D_{JS}} \downarrow$
Exact erasure	– *	0.27
Sundararajan et al. (2017)	1.32	0.27
Schulz et al. (2020)	1.12	0.18
Guan et al. (2019)	0.88	0.24
DiffMask	**0.01**	**0.00**

Table 1: Toy task: attribution to hidden states, average divergence in nats between the *ground-truth* attributions and those by different methods. *The Delta distribution does not share support with the *ground-truth*.

plied. Once we have established that DiffMask can be trusted, we use it to analyze BERT-based models (Devlin et al., 2019) fine-tuned on sentiment classification, and on question answering. We report hyperparameters in Appendix C, and additional plots, examples and analysis in Appendix D.

3.1 Toy task

Our toy task is defined as: given a sequence x of digits (i.e., $x_i \in \{0, \cdots, 9\}$), and a query $\langle n, m\rangle$ of two digits, determine whether $\#n > \#m$ in x.

Model The query and input are embedded, concatenated, and then fed to a single-layer feed-forward NN, followed by a single-layer unidirectional GRU (Cho et al., 2014).[3] The classification is done by a linear layer that acts on the last hidden state of the GRU. See Appendix C.1 for all hyperparameters and a more precise definition of the architecture. Unsurprisingly, the model solves the task almost perfectly (accuracy on test is $>99\%$).

Ground-truth for hidden-state attribution We plot the distribution of hidden states (we use dimensionality 2, with the purpose of having a bottleneck and to support clear visualization) and observe a linear separation between states of digits present in the query and states not in the query. This means that the role of the feed-forward layer is to decide which digits to keep. Since the model solves the task, the role of the GRU must then be to count which digit occurred the most. The prediction must be attributed uniformly to *all* the hidden states corresponding to either n or m. For completeness, Figure 11 in the Appendix D.1 shows this plot.

Results We start with an example of *input attributions*, see Figure 3, which illustrates how DiffMask goes beyond input attribution as typically known.[4] The attribution provided by erasure (Figure 3a) is not informative: for each datapoint the search always finds a single digit that is sufficient to maintain the original prediction and discards all the other inputs. The perturbation methods by Schulz et al. (2020) and Guan et al. (2019) (Figure 3b and 3d) are also over-aggressive in pruning. They assign low attribution to some items in the query even though those had to be considered when making the prediction. Differently from other methods, DiffMask reveals input attributions conditioned on different levels of depth. Figure 3e shows both input attributions according to the input itself and according to the hidden layer. It reveals that at the embedding layer there is no information regarding what part of the input can be erased: attribution is uniform over the input sequence. After the model has observed the query, hidden states predict that masking input digits other than n and m will not affect the final prediction: attribution is uniform over digits in the query. This reveals the role of the feed-forward layer as a filter for positions relevant to the query. Other methods do not allow for this type of inspection. These observations are consistent across the entire test set.

For *attribution to hidden states* (i.e., the output of the feed-forward layer) we can compare methods in terms of how much their attributions resemble the ground-truth across the test set. Table 1 shows how the different approaches deviate from the gold-truth in terms of Kullback-Leibler ($\mathrm{D_{KL}}$) and Jensen–Shannon ($\mathrm{D_{JS}}$) divergences.[5]

3.2 Sentiment Classification

We turn now to a real task and analyze models fine-tuned for sentiment classification on the Stanford Sentiment Treebank (SST; Socher et al., 2013).

Erasure search as learning masks Before diving into an analysis of a BERT sentiment model, we would like to demonstrate that we can approximate the result of erasure well through our differentiable relaxations. For that, we train a single-layer GRU sentiment classifier and compare the analyses by DiffMask to solutions provided by

[3] We use a feed-forward NN to incorporate the query information, rather than another GRU layer, to ensure that counting cannot happen in the first layer. This helps us define the ground-truth for the method.

[4] To enable comparison across methods, the attributions in this Section are normalized between 0 and 1.

[5] We use $\mathrm{D_{KL}}[p||q]$ and $\mathrm{D_{JS}}[p||q]$ where p is the *ground-truth* distribution and q is the predicted attribution distribution.

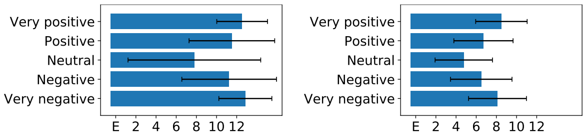
Metric	REINFORCE+	DIFFMASK
Precision	74.69	**81.26**
Recall	80.82	**85.89**
F_1	73.57	**80.75**
Optimality	8.83	**32.67**
L_0	33.13	**30.58**

Table 2: Sentiment classification: optimization with DIFFMASK and REINFORCE (not amortised – with a moving average baseline for variance reduction) vs. erasure with exact search. All metrics are computed at token level; optimality is measured at sentence level.

erasure (exact search). To isolate the impact of our objective, we disable amortization, thus estimating Hard Concrete parameters for each example independently. We compare DIFFMASK to REINFORCE (Williams, 1992) with a moving average baseline for variance reduction. Since erasure is prohibitive for long sentences, we limit our evaluation to sentences up to 25 words (54% of the data). Table 2 shows that DIFFMASK and REINFORCE achieve comparable levels of sparsity, but our method reaches an optimal solution much more often (33% of the times vs 9%) and is, on average, closer to an optimal solution (81% F_1 vs 75% F_1).

Faithfulness and Plausibility Now, we get back to the fully-amortized DIFFMASK approach applied to a 12-layers $\text{BERT}_{\text{BASE}}$ model and verify that there is no performance degradation when applying masking. Training hyperparameters are reported in Appendix C.2. The F_1 score of the model on the validation set moved from 37.9% to 38.3% while masking 46.3% input tokens, and to 38.9% while masking 67.6% hidden states. The explanations provided by DIFFMASK are also stable. Across 5 runs with different seeds, the standard deviation of input attributions are 0.05 and 0.03 for inputs and hidden states, respectively.

While we cannot use human labels to evaluate faithfulness of our method, comparing them and DIFFMASK attribution will tell us whether the sentiment model relies on the same cues as humans. Specifically, we compare to SST token level annotation of sentiment. In Figure 4a, we show after how many layers on average an input token is dropped, depending on its sentiment label. This suggests that the model relies more heavily on strongly positive or negative words and, thus, is generally consistent with human judgments (i.e., *plausible*).



(a) Input.

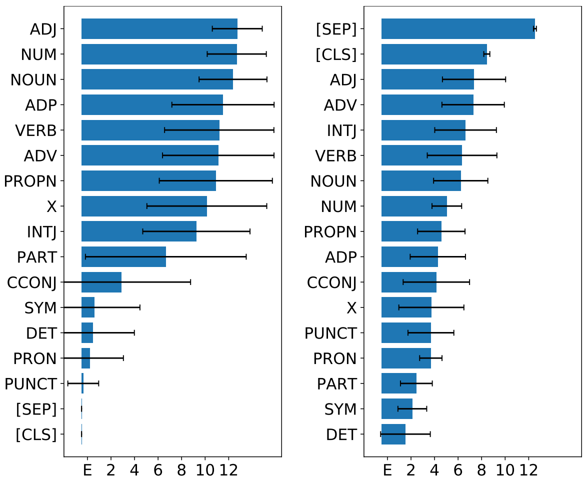
(b) Hidden states.

Figure 4: Sentiment classification: average number of layers that predict to keep input tokens or hidden states aggregated by token level sentiment annotations.

Analysis We used DIFFMASK to analyse the behavior of our BERT model. In Figure 5, we report the average number of layers that input tokens or hidden states are kept for (or, equivalently, after how many layers they are dropped on average), aggregating by part-of-speech tags (PoS). It turns out that determinants, punctuation, and pronouns can be completely discarded from the input across all validation set, while adjectives and nouns should be kept. Also the `[CLS]` and `[SEP]` tokens can be ignored indicating that the model does not need such markers. Examining the POS tags distribution for hidden states leads to further conclusions. Here, the `[CLS]` and `[SEP]` tokens are the most important ones. This is not surprising as the classifier on top of BERT uses the `[CLS]` hidden state which gets progressively updated through all layers. Both these special tokens are not important as inputs because BERT can infer these markers in other layers, however, they are heavily used in the computation.

Figure 6e we show a visual example of that. We see that the model, even in the bottom layers, knows that the punctuation and both separators can be dropped from the input. This contrasts with hidden states attribution (Figure 6f) which indicates that the separator states (especially `[SEP]`) are very important. By putting this information together, we can hypothesize that the separator is used to aggregate information from the sentence, relying on self-attention. In fact, this aggregation is still happening in layer 12; at the very top layers, states corresponding to almost all non-separator tokens can be dropped.

Comparison to other methods In Figure 6, we visually compare different techniques on one example form validation set. While previous techniques (e.g., integrated gradient) do not let us test what a model 'knows' in a given layer (i.e. attribution to input conditioned on a layer), they can be used to perform attribution to hidden layers. All methods except attention correctly highlight the last hidden



(a) Inputs.

(b) Hidden states.

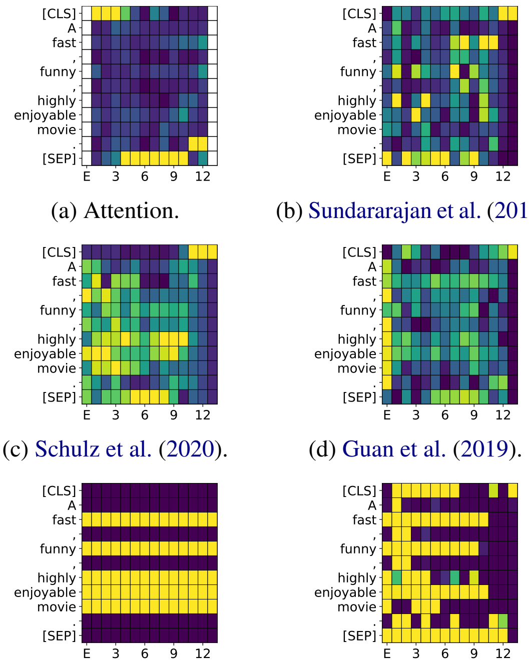
Figure 5: Sentiment classification: average number of layers that predict to keep input tokens (a) or hidden states (b) aggregating by part-of-speech tags (POS) and [CLS], [SEP] tokens on validation set.

state of the [CLS] token as important. Its importance is due to the top-level classifier using the [CLS] hidden state. Although for DIFFMASK we show the expectation of keeping states, it assigns much sharper attributions. For instance, on the validation set, it assigns to the last hidden state of the [CLS] the biggest attribution 99% of the times where Schulz et al. (2020) only 71%. Raw attention (Figure 6a) does not seem to highlight any significant patterns in that example except that start and end of sentence tokens ([CLS] and [SEP], respectively) receive more attention than the rest.[6] Attributions by Schulz et al. (2020) and Guan et al. (2019) assign slightly higher importance to hidden states corresponding to 'highly' and 'enjoyable', whereas it is hard to see any informative patterns provided by integrated gradient. Notice that for DIFFMASK, a near-zero attribution has a very clear interpretation: such a state is not used for prediction since in expectation it is dropped (not gated).

3.3 Question Answering

We turn now to QA where we analyse a fine-tuned BERT$_{\text{LARGE}}$ model on the Stanford Question Answering Dataset (SQuAD; Rajpurkar et al., 2016).

Analysis We start by asking DIFFMASK **which tokens does the model keep?** We do a similar analysis as for sentiment classification of POS tags over the entire validation set. We summarize the



(a) Attention.

(b) Sundararajan et al. (2017).

(c) Schulz et al. (2020).

(d) Guan et al. (2019).

(e) DIFFMASK on input.

(f) DIFFMASK on hidden.

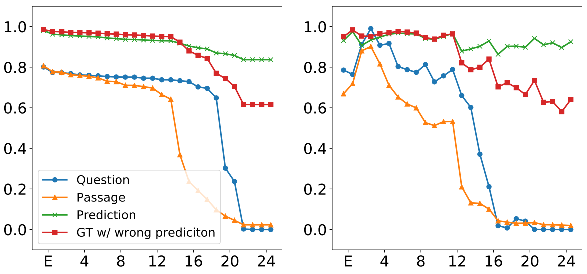
Figure 6: Sentiment classification: comparison between attribution method for hidden layers w.r.t. the predicted label. All plots are normalized per-layer by the largest attribution. Attention heatmap is obtained max pooling over heads and averaging across positions.

results in Figure 14 in Appendix D.2. It turns out that conjunctions and adpositions are dropped by the embedding and first layer, respectively, on average. On the contrary, proper nouns and punctuation are usually predicted to be dropped only after the 14th layer. We argue that due to the pre-training objective, BERT could infer well missing parts of the input, especially if they are trivial to infer (e.g., as often the case for prepositions). On the contrary, nouns and proper nouns are important as they count for 84% of the answers on SQuAD. For example, in Figure 8a, we can see that it takes 13–16 layers for the model to 'realize' that *'Santa Clara Marriot'* is not relevant to the question and discard it.

Unlike in sentiment classification, separator tokens as well as punctuation assume a central role as inputs (i.e., punctuation is considered the most important POS tag as for both questions and passages is usually dropped after the 17th layer). Punctuation serves to demarcate sentence boundaries, useful for QA but not for sentiment classification.

Tokens from questions are generally masked by higher layers than tokens from passages as we show in Figure 7a, which suggests that they are more important. We highlight that even in higher

[6] Voita et al. (2019b) and Michel et al. (2019) pointed out that many Transformer heads play no or minor role.



(a) Input.

(b) Hidden states.

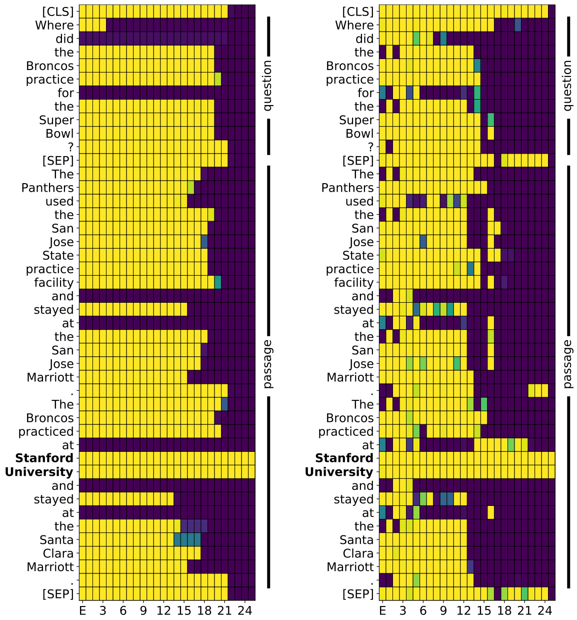
Figure 7: QA: average expectation of keeping input (a) and hidden states (b) from different layers.

layers when DiffMask masks $> 95\%$ of the tokens, the original model prediction is almost always kept $> 90\%$. Noticeably, when the original BERT makes wrong predictions, the tokens annotated as the ground truth answer are kept $\sim 60\%$ of the time. This may suggest that when this happens the model still considers other options (e.g., valid options such as the ground truth) as plausible, thus DiffMask detects them as important.

Now, we inspect hidden states attributions to answer **where is the information stored?** In Figure 7b we can see a similar trend as for masking input, i.e., question's hidden states are kept more on average and deeper in the computation. States on layers 2–3 are dropped less than from the embedding and first layer. This is consistent with findings of Voita et al. (2019a) which show that frequent tokens, such as determiners, accumulate contextual information. However, they are not important as inputs as we show in an example in Figure 8b.

The hidden states corresponding to separator tokens are always kept across all layers except the last one across the validation set. Notice that, this token is also used as a delimiter between the question and the passage, and hence indicates where questions as well as passages end.

The level of hidden states pruning is quite incremental (after layer 3) and gets strong, after layer 9 more than 50% of them can be masked out. A steep increase in superfluous states 13–14 (visible on both parts of Figure 7) may indicate that some states, at that point in computation, contain enough information needed for the classification while all the others can indeed be removed without affecting the model prediction. Our observation that higher layers are more predictive is in line with findings of Kovaleva et al. (2019). They pointed out that the final layers of BERT change most and are more task-specific. Again, the fact that states corresponding to the ground truth answer are still active on top layers when the model makes a wrong prediction indicates that the model is still considering different span options across top layers as well.



(a) Gating the input.

(b) Gating hidden states.

Figure 8: QA: attribution the inputs (a) and hidden states (b). The correct answers is highlighted in bold.

Comparison to other methods As we do not have access to the ground-truth, we start by contrasting DiffMask qualitatively to other attribution methods on a few examples. We highlight some common pitfalls that afflict other methods (such as the hindsight bias) and how DiffMask overcomes those. This helps demonstrate our method's faithfulness to the original model.

Figure 2 shows input attributions by different methods on an example from the validation set. Erasure (Figure 2d), as expected, does not provide useful insights, it essentially singles out the answer discarding everything else including the question. This cannot be faithful and is a simple consequence of erasure's hindsight bias: when only the span that contains the answer is presented as input, the model predicts that very span as the answer, but this does not imply that the model ignores everything else when presented with the complete document as input. The methods of Schulz et al. (2020) and Guan et al. (2019) optimize attributions on single examples and thus also converge to assigning high importance mostly to words that support the current prediction and that indicate the question type. Integrated gradient does not seem to highlight any discernible pattern, which we speculate is mainly because a zero baseline is not suitable for word embeddings. Choosing a more adequate baseline is not straightforward and remains an important open issue (Sturmfels et al., 2020). Note that, DIFFMASK without amortization (Figure 2f) resembles erasure (as shown in § 3.2 for SST).

Differently from all other methods, our DIFFMASK probes the network to understand what it 'knows' about the input-output mapping in different layers. In Figure 2e we show the expectation of keeping input tokens conditioned on any one of the layers in the model to make such predictions (see Figure 8a for a per-layer visualization). Our input attributions highlight that the model, in expectation across layers, *wants* to keep words in the question, the predicate 'practice' in both sentences as well as all potential candidate answers (i.e., named entities). But eventually, the most important spans are in the question and the answer itself.

4 Related Work

While we motivated our approach through its relation to erasure, an alternative way of looking at our approach is considering it as a *perturbation-based* method. This recently introduced class of attribution methods (Ying et al., 2019; Guan et al., 2019; Schulz et al., 2020; Taghanaki et al., 2019), instead of erasing input, injects noise. Besides back-propagation and attention-based methods discussed in the introduction, another class of interpretation methods (Murdoch and Szlam, 2017; Singh et al., 2019; Jin et al., 2020) builds on prior work in cooperative game theory (e.g., Shapley value of Shapley, 1953). These methods are not trivial to apply to a new model, as they are architecture-specific. Their hierarchical versions (e.g., Singh et al., 2019; Jin et al., 2020) also make a strong assumption about the structure of interaction (e.g., forming a tree) which may affect their faithfulness. Also Chen et al. (2018) share some similarities to our work as they also do amortization but use the *Gumbel softmax trick* (Maddison et al., 2017; Jang et al., 2017) to approximate minimal subset selection. They assume that the subset contains exactly k elements where k is a hyperparameter. Moreover, their explainer is a separate model predicting input subsets, rather than a 'probe' on top of the model's hidden layers, and hence cannot be used to reveal how decisions are formed across layers.

A large body of literature analyzed BERT and Transformed-based models. For example, Tenney et al. (2019) and van Aken et al. (2019) probed BERT layers for a range of linguistic tasks, while Hao et al. (2019) analyzed the optimization surface. Rogers et al. (2020) provides a comprehensive overview of recent BERT analysis papers.

There is a stream of work on learning interpretable models by means of extracting latent rationales (Lei et al., 2016; Bastings et al., 2019). Some of the techniques underlying DIFFMASK are related to that line of work. They employ stochastic masks to *learn an interpretable model*, which they train by minimizing a downstream loss subject to constraints on L_0, whereas we employ stochastic masks to *interpret an existing model*, and for that, we minimize L_0 subject to constraints on that model's output distribution. In our very recent work Schlichtkrull et al. (2020), we also employ stochastic masks and L_0 regularization for analyzing graph neural networks. We learn which edges are relevant in multi-hop question answering and graph-based semantic role labeling (Marcheggiani and Titov, 2017; De Cao et al., 2019).

5 Conclusion

We have introduced a new *post hoc* interpretation method which learns to completely remove subsets of inputs or hidden states through masking. We circumvent an intractable search by learning an end-to-end differentiable prediction model. To overcome the hindsight bias problem, we probe the model's hidden states at different depths and amortize predictions over the training set. Faithfulness is validated in a controlled experiment pointing more clearly to some flaws of other attribution methods. We used our method to study BERT-based models on sentiment classification and question answering. DIFFMASK sheds light on what different layers 'know' about the input and where information about the prediction is stored in different layers.

Acknowledgements

The authors want to thank Christos Baziotis, Elena Voita, Dieuwke Hupkes, and Naomi Saphra for helpful discussions. This project is supported by SAP Innovation Center Network, ERC Starting Grant BroadSem (678254), the Dutch Organization for Scientific Research (NWO) VIDI 639.022.518, and the European Union's Horizon 2020 research and innovation programme under grant agreement No 825299 (Gourmet).

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *International Conference on Learning Representations*.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A Gers. 2019. How Does BERT Answer Questions? A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7).

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. 2010. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-SNLI: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9539–9549.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. 2018. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 883–892, Stockholmsmässan, Stockholm Sweden. PMLR.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2306–2317, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.

Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309.

Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. 2019. Towards a deep and unified understanding of deep neural models in NLP. In *International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2454–2463, Long Beach, California, USA. PMLR.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and understanding the effectiveness of BERT. 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 4143–4152, Hong Kong, China. Association for Computational Linguistics.

Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé, Miro Dudik, and Hanna Wallach. 2019. Improving fairness in machine learning systems: What do industry practitioners need? In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–16, New York, NY, USA. Association for Computing Machinery.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-Softmax. *International Conference on Learning Representations*.

Xisen Jin, Junyi Du, Zhongyu Wei, Xiangyang Xue, and Xiang Ren. 2020. Towards Hierarchical Importance Attribution: Explaining Compositional Semantics for Neural Sequence Models. *International Conference on Learning Representations*.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Been Kim. 2015. *Interactive and interpretable machine learning models for human machine collaboration*. Ph.D. thesis, Massachusetts Institute of Technology.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *International Conference on Learning Representations*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. 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 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv:1612.08220*.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning Sparse Neural Networks through L_0 Regularization. In *International Conference on Learning Representations*.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. *International Conference on Learning Representations*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1507–1516, Copenhagen, Denmark. Association for Computational Linguistics.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14014–14024. Curran Associates, Inc.

W James Murdoch and Arthur Szlam. 2017. Automatic rule extraction from long short term memory networks. *International Conference on Learning Representations*.

Weili Nie, Yang Zhang, and Ankit Patel. 2018. A theoretical explanation for perplexing behaviors of backpropagation-based visualizations. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3809–3818, Stockholmsmässan, Stockholm Sweden. PMLR.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China. PMLR.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the*

Association for Computational Linguistics: Demonstrations, pages 97–101, San Diego, California. Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. *International Conference on Learning Representations (ICLR)*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What we know about how BERT works. *arXiv:2002.12327*.

Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. 2020. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. *arXiv:2010.00577*.

Karl Schulz, Leon Sixt, Federico Tombari, and Tim Landgraf. 2020. Restricting the flow: Information bottlenecks for attribution. In *International Conference on Learning Representations*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Lloyd S Shapley. 1953. *A value for n-person games*, volume 2, pages 307–317. Princeton University Press, Princeton.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153, International Convention Centre, Sydney, Australia. PMLR.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. *Workshop at International Conference on Learning Representations*.

Chandan Singh, W. James Murdoch, and Bin Yu. 2019. Hierarchical interpretations for neural network predictions. In *International Conference on Learning Representations*.

Leon Sixt, Maximilian Granz, and Tim Landgraf. 2020. When Explanations Lie: Why Many Modified BP Attributions Fail. *International Conference on Machine Learning*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA. Association for Computational Linguistics.

Pascal Sturmfels, Scott Lundberg, and Su-In Lee. 2020. Visualizing the impact of feature attribution baselines. *Distill*. https://distill.pub/2020/attribution-baselines.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328, International Convention Centre, Sydney, Australia. PMLR.

Saeid Asgari Taghanaki, Mohammad Havaei, Tess Berthier, Francis Dutil, Lisa Di Jorio, Ghassan Hamarneh, and Yoshua Bengio. 2019. Infomask: Masked variational latent representation to localize chest disease. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, pages 739–747, Cham. Springer International Publishing.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across NLP tasks. *arXiv:1909.11218*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019a. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. 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 4396–4406, Hong Kong, China. Association for Computational Linguistics.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019b. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.

Thomas Wolf, L Debut, V Sanh, J Chaumond, C Delangue, A Moi, P Cistac, T Rault, R Louf, M Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv:1910.03771*.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 9244–9255. Curran Associates, Inc.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. 2019. Lookahead optimizer: k steps forward, 1 step back. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 9597–9608. Curran Associates, Inc.

Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. 2017. Visualizing deep neural network decisions: Prediction difference analysis. *International Conference on Learning Representations*.

A Probe parameterization

We parameterized the probe functions with a single layer MLP. Note that the architecture of this probe is chosen to be simple but different model choices are also possible and will not affect our general framework.[7] When masking input tokens, 'votes' are computed as $v_i^{(\ell)} = g_\phi^{(\ell)}(h_i^{(\ell)})$ where

$$\gamma_i^{(\ell)} = \xi \cdot \tanh\left(\mathrm{NN}^{(\ell)}([x_i; h_i^{(\ell)}])\right) + b^{(\ell)} \ , \quad (4)$$

$$v_i^{(\ell)} \sim \mathrm{HardConcrete}(v_i^{(\ell)}; \tau, \gamma_i^{(\ell)}, l, r) \ , \quad (5)$$

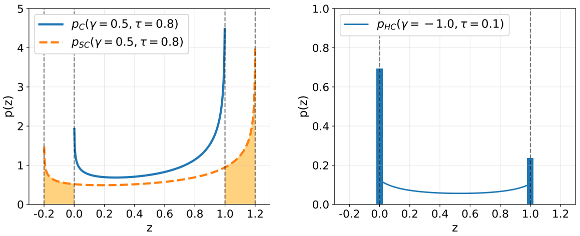
where $\xi = 10, \tau = 0.2, l = -0.2, r = 1.0$ are fixed hyperparameters. See Appendix B for details about the Hard Concrete distribution including its parameterization. NN are feed-forward neural networks with architecture $[H/4, \tanh, 1]$ where H is the BERT hidden size, bs are learnable biases. We use the same functional form to compute $z^{(\ell)}$ (masking hidden states) but x_i omitted from the input of the feed-forward NN. For the input probe the output of the last projection (but not the bias) is constrained to be $\in (-\xi, \xi)$ for numerical stability. We initialized the bias of the last FFNN layer to 5 to start with high probability of keeping states (fundamental for good convergence as the initialized DIFFMASK has not learned what to mask yet).

B The Hard Concrete distribution

The Hard Concrete distribution, assigns density to continuous outcomes in the open interval $(0, 1)$ and non-zero mass to exactly 0 and exactly 1. A particularly appealing property of this distribution is that sampling can be done via a differentiable reparameterization (Rezende et al., 2014; Kingma and Welling, 2014). In this way, the $\mathcal{L}_0$ loss in Equation 1 becomes an expectation

$$\mathcal{L}_0(\phi, b|x) = \sum_{i=1}^{N} \mathbb{E}_{p_\phi(z_i|x)}\left[z_i \neq 0\right] \ . \quad (6)$$

whose gradient can be estimated via Monte Carlo sampling without the need for REINFORCE and without introducing biases. We did modify the original Hard Concrete, though only so slightly, in a way that it gives support to samples in the half-open interval $[0, 1)$, that is, with non-zero mass only at 0. That is because we need only distinguish 0 from non-zero, and the value 1 is not particularly important.[8]



(a) (b)

Figure 9: Binary Concrete distributions: (a) a Concrete p_C and its stretched version p_{SC}; (b) a rectified and stretched (Hard) Concrete p_{HC}.

The distribution A stretched and rectified Binary Concrete (also known as Hard Concrete) distribution is obtained applying an affine transformation to the Binary Concrete distribution (Maddison et al., 2017; Jang et al., 2017) and rectifying its samples in the interval $[0, 1]$ (see Figure 9). A Binary Concrete is defined over the open interval $(0, 1)$ (p_C in Figure 9a) and it is parameterised by a location parameter $\gamma \in \mathbb{R}$ and temperature parameter $\tau \in \mathbb{R}_{>0}$. The location acts as a logit and it controls the probability mass skewing the distribution towards 0 in case of negative location and towards 1 in case of positive location. The temperature parameter controls the concentration of the distribution. The Binary Concrete is then stretched with an affine transformation extending its support to (l, r) with $l \leq 0$ and $r \geq 1$ (p_{SC} in Figure 9a). Finally, we obtain a Hard Concrete distribution rectifying samples in the interval $[0, 1]$. This corresponds to collapsing the probability mass over the interval $(l, 0]$ to 0, and the mass over the interval $[1, r)$ to 1 (p_{HC} in Figure 9b). This induces a distribution over the close interval $[0, 1]$ with non-zero mass at 0 and 1. Samples are obtained using

$$\begin{aligned} s &= \sigma\left(\left(\log u - \log(1-u) + \gamma\right)/\tau\right) \ , \\ z &= \min\left(1, \max\left(0, s \cdot (l - r) + r\right)\right) \ , \end{aligned} \quad (7)$$

where σ is the Sigmoid function $\sigma(x) = (1 + e^{-x})^{-1}$ and $u \sim \mathcal{U}(0, 1)$. We point to the Appendix B of Louizos et al. (2018) for more information about the density of the resulting distribution and its cumulative density function.

[7] In our open source implementation, we also used different architectures. Final results did not change much.

[8] Only a true 0 is guaranteed to completely mask an input out, while any non-zero value, however small, may leak some amount of information.

Latent rationales There is a stream of work on learning interpretable models by means of extracting latent rationales (Lei et al., 2016; Bastings et al., 2019). Some of the techniques underlying DiffMask are related to that line of work, but overall we approach very different problems. Lei et al. (2016) use REINFORCE to minimize a downstream loss computed on masked inputs, where the masks are binary and latent. They employ L_0 regularization to solve the task while conditioning only on small subsets of the input regarded as a *rationale* for the prediction. To the same end, Bastings et al. (2019) minimize downstream loss subject to constraints on expected L_0 using a variant of the sparse relaxation of Louizos et al. (2018). In sum, they employ stochastic masks to learn an interpretable model which they learn by minimizing a downstream loss subject to constraints on L_0, we employ stochastic masks to interpret an existing model and for that we minimize L_0 subject to constraints on that model's downstream performance.

C Hyperparameters

C.1 Toy task

Data We generate sequences of varying length (up to 10 digits long) sampling each element independently: with 50% probability, we draw uniformly n or m and, with 50% probability, we draw uniformly from the remaining digits. We generate 10k data-points, keeping 10% of them for validation. The space of input sequences is $> 10^{10}$. Thus, a model that solves the task cannot simply memorize the training set.

Model The precise model formulation is the following: given a query $q = \langle n, m \rangle$ and an input $x = \langle x_1, \dots x_t \rangle$, they are embedded as

$$\begin{aligned} n' &= \mathrm{Emb}_q(n) \, , \\ m' &= \mathrm{Emb}_q(m) \, , \\ x'_i &= \mathrm{Emb}_x(x_i) \qquad \forall i \in 1 \dots t \, , \end{aligned} \tag{8}$$

where Emb_q and Emb_x are embedding layers of dimensionality 64. The prediction is computed as

$$\begin{aligned} h_i^{(1)} &= \mathrm{FFNN}([n'; m'; x'_i]) \quad \forall i \in 1 \dots t \, , \\ h_0^{(2)} &= [0 \dots 0]^\top \, , \\ h_i^{(2)} &= \mathrm{GRU}(h_i^{(1)}, h_{i-1}^{(2)}) \qquad \forall i \in 1 \dots t \, , \\ y &= w^\top h_t^{(2)} + b \, , \end{aligned} \tag{9}$$

where $[\cdot;\cdot]$ denotes concatenation, FFNN is a feed-forward neural network with architecture $[64 \times 3, \tanh, 2]$, GRU is a Gated Recurrent Network (Cho et al., 2014) with hidden size of 64, and $w \in \mathbb{R}^{64}$, $b \in \mathbb{R}$ are the weight and bias parameter of the final classifier respectively.

Model	**Value**
Type	BERT$_{\text{BASE}}$ (uncased)
Layers	12
Hidden units	768
Pre-trained masking	standard
Optimizer	Adam*
Learning rate	$3 \cdot 10^{-5}$
Train epochs	50
Batch size	64
DiffMask	**Value**
Optimizer	Lookahead RMSprop**
Learning rate ϕ, b	$3 \cdot 10^{-4}$
Learning rate λ	$1 \cdot 10^{-1}$
Train epochs	100
Batch size	64
Constrain	$\mathrm{D_{KL}}[y \| \hat{y}] < 0.5$

Table 3: Hyperparameters for the sentiment classification experiment. Optimizers: * Kingma and Ba (2015), ** Tieleman and Hinton (2012); Zhang et al. (2019).

Attribution methods Integrated gradient attribution (Sundararajan et al., 2017) is computed with 500 steps. Attribution of Schulz et al. (2020) is computed at token level with $\beta = 10/k$ where k is the token embedding size. We optimized using the RMSprop (Tieleman and Hinton, 2012) with learning rate 10^{-1} for 500 steps. Attribution of Guan et al. (2019) is computed at token level with $\lambda = 10^{-4}$ using RMSprop with learning rate 10^{-1} for 500 steps. Our DiffMask is optimized for 100 epochs using Lookahead RMSprop (Tieleman and Hinton, 2012; Zhang et al., 2019) with learning rate 10^{-2} for ϕ, b and 10^{-1} for α. For these attribution methods we used our own re-implementation.

C.2 Sentiment Classification

Data We used the Stanford Sentiment Treebank (SST; Socher et al., 2013) available here[9]. We pre-processed the data as in Bastings et al. (2019). Training and validation sets contain 8544 and 1101 sentences respectively.

[9] https://nlp.stanford.edu/sentiment/trainDevTestTrees_PTB.zip

Model	**Value**
Type	BERT$_{\text{LARGE}}$ (uncased)
Layers	24
Hidden units	1024
Pre-trained masking	whole-word
Optimizer	Adam*
Learning rate	$3 \cdot 10^{-5}$
Train epochs	2
Batch size	24
DiffMask	**Value**
Optimizer	Lookahead RMSprop**
Learning rate ϕ, b	$3 \cdot 10^{-4}$
Learning rate λ	$1 \cdot 10^{-1}$
Epochs (inputs)	1 (per layer)
Epochs (hidden)	4
Batch size	8
Constrain	$\mathrm{D}_{\mathrm{KL}}[y \| \hat{y}] < 1$

Table 4: Hyperparameters for the question answering experiment. Optimizers: * Kingma and Ba (2015), ** Tieleman and Hinton (2012); Zhang et al. (2019).

Model For the sentiment classification experiment we downloaded[10] a pre-trained model from the Huggingface implementation[11] of Wolf et al. (2019), and we fined-tuned on the SST dataset. We report hyperparameters used for training the model and our DiffMask in Table 3.

C.3 Question Answering

Data We used the Stanford Question Answering Dataset (SQuAD v1.1; Rajpurkar et al., 2016) available here[12]. Pre-processing excluded QA pairs with more than 384 BPE tokens to avoid memory issues. After this we end up having 86706 training instances and 10387 validation instances.

Model For the question answering experiment we downloaded [10] an already fine-tuned model from the Huggingface implementation[11] of Wolf et al. (2019) We report hyperparameters used by them for training the original model and the ones used for our DiffMask in Table 4.

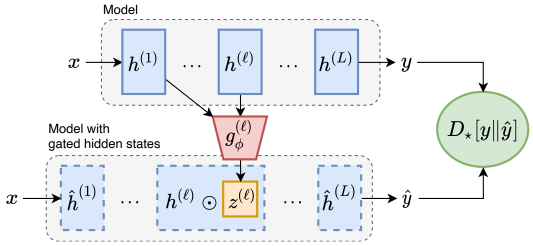


Figure 10: DiffMask for hidden states: states up to layer ℓ from a model (top) are fed to a classifier g that predicts a mask $z^{(\ell)}$. We use this to mask the ℓih hidden state and re-compute the forward pass from that point on (bottom). The classifier g is trained to mask the hidden state as much as possible without changing the output (minimizing a divergence D$_\star$).

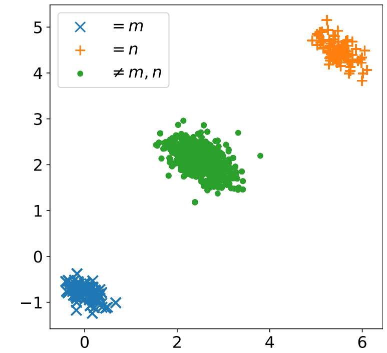


Figure 11: Hidden state values for the two-neuron toy task. Clusters of whether the input digit is equal to the first or second position in the query ($= n$ or $= m$ respectively) or not at all ($\neq n, m$) are completely linear separable.

D Additional plots and results

In Figure 10 we show an overview of the variant of DiffMask to analyze the hidden states of a model (see Figure 1 to compare the two versions).

D.1 Toy task

In Figure 11 we show the distribution of hidden states in the toy task where we highlight whether they belong to a state corresponding to n, m or neither of them.

D.2 Sentiment Classification

In Figure 13 we show an additional comparison example between attribution method for hidden layers w.r.t the predicted label.

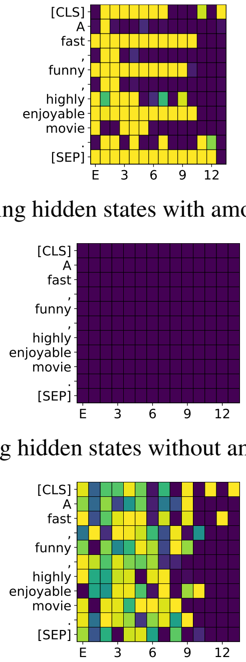
D.2.1 Ablation

As argued in the introduction and shown on the toy task, many popular methods (e.g., erasure and its approximations) are over-aggressive in discarding

[10] https://huggingface.co/transformers/pretrained_models.html

[11] https://github.com/huggingface/transformers

[12] https://rajpurkar.github.io/SQuAD-explorer



(a) Masking hidden states with amortization.

(b) Masking hidden states without amortization.

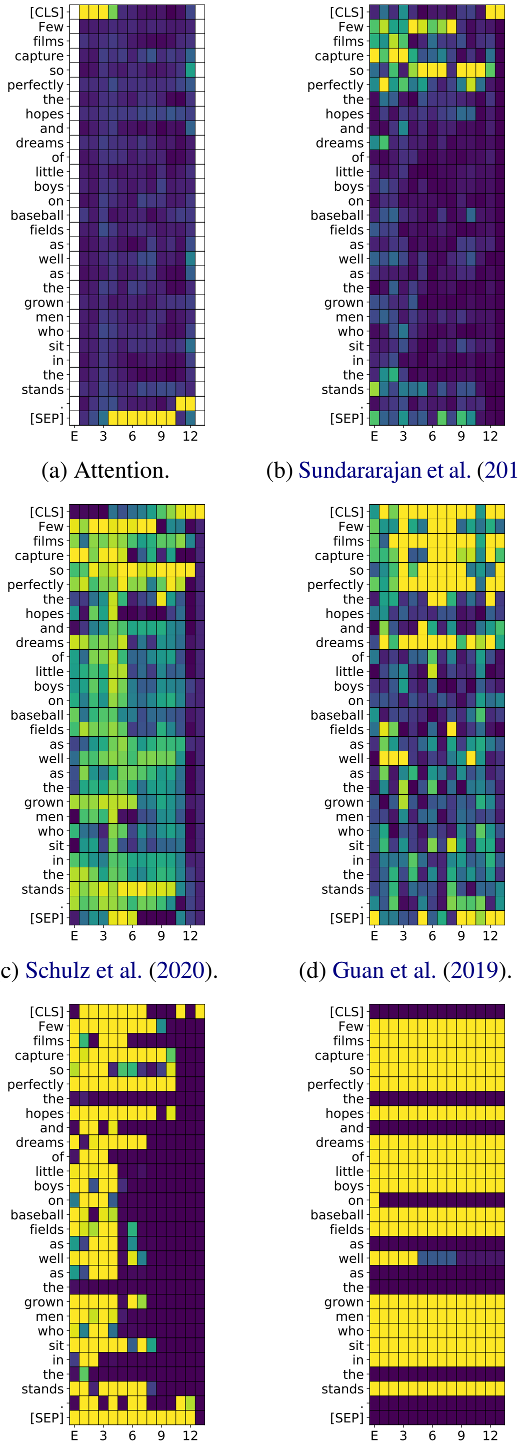
(c) Masking hidden states without amortization and without baseline.

Figure 12: Sentiment classification: ablation study on amortization and baseline.

inputs and hidden units. Amortization is a fundamental component of DIFFMASK and is aimed at addressing this issue. In Figure 12 we show how our method behaves when ablating amortization and thus optimizing on a single example instead. Noticeable, our method converges to masking out all hidden states at any layer (Figure 12b). This happens as it learns an *ad hoc* baseline just for that example. When we ablate both amortization and baseline learning (Figure 12c), the method struggles to uncover any meaningful patterns. This highlights how both core components of our method are needed in combination with each other.

D.3 Question Answering

In Figure 14 we report statistics on the average number of layers that predict to keep input tokens aggregating by POS tag. We report additional two examples of expectation predicted by DIFFMASK in Figure 15.



(a) Attention.

(b) Sundararajan et al. (2017).

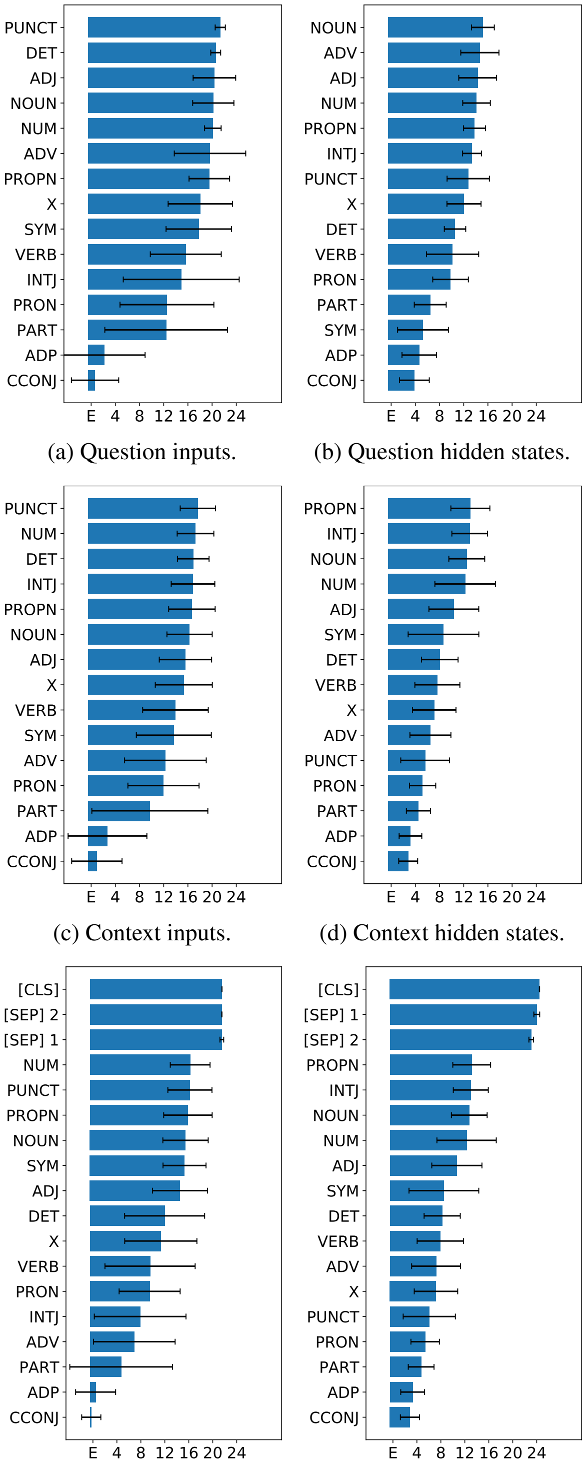
(c) Schulz et al. (2020).

(d) Guan et al. (2019).

(e) DIFFMASK.

(f) DIFFMASK on input.

Figure 13: Sentiment classification: comparison between attribution method for hidden layers w.r.t. the predicted label. All plots are normalized per-layer by the largest attribution. Attention heatmap is obtained max pooling over heads and averaging across positions.



(a) Question inputs.

(b) Question hidden states.

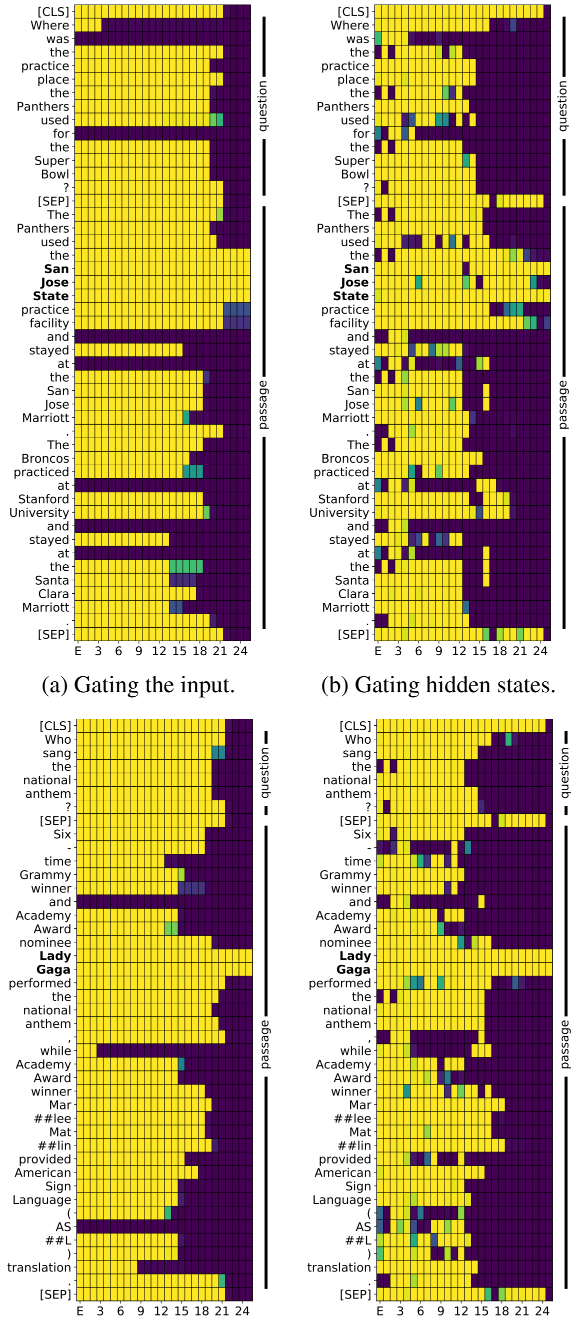
(c) Context inputs.

(d) Context hidden states.

(e) All inputs.

(f) All hidden states.

Figure 14: Question answering: average number of layers that predict to keep input tokens (a), (c) and (e) or hidden states (b), (d) and (f) aggregating by part-of-speech tag (POS) on validation set.



(a) Gating the input.

(b) Gating hidden states.

(c) Gating the input.

(d) Gating hidden states.

Figure 15: Expectation predicted by DIFFMASK to keep the inputs in (a) (c) and hidden states in (b) (d) on two different QA pairs. The correct answers is highlighted in bold.