Decoding Layer Saliency in Language Transformers

Elizabeth M. Hou [1] **Gregory Castanon** [1]

Abstract

In this paper, we introduce a strategy for identifying textual saliency in large-scale language models applied to classification tasks. In visual networks where saliency is more well-studied, saliency is naturally localized through the convolutional layers of the network; however, the same is not true in modern transformer-stack networks used to process natural language. We adapt gradient-based saliency methods for these networks, propose a method for evaluating the degree of semantic coherence of each layer, and demonstrate consistent improvement over numerous other methods for textual saliency on multiple benchmark classification datasets. Our approach requires no additional training or access to labelled data, and is comparatively very computationally efficient.

1. Introduction

Trained on the vast swathes of open-source text available on the internet, large-scale language models have demonstrated impressive performance in text generation and classification. Most recently, models with transformer-stack architectures have shown an impressive ability to focus on task-salient elements of language and utilize that focus to achieve superhuman performance in certain constrained areas. However, there is a growing concern that despite this performance, these models lack transparency and have unpredictable blind spots in certain areas. This has led to an increased focus on *salience* in natural language i.e. identifying which elements of text the model considers important for making a decision.

Unlike computer vision, where the pixels relevant to a task are often grouped together, the words that are important in a movie review, article or resume may not be close to each other. This lack of locality is reflected in the preferred architectures, with convolutional heads to visual networks encouraging local associations while stacks of fully connected transformer layers allow natural language tokens to associate more globally. This free association, combined with the degree of model complexity in transformer architectures, leads to challenges in interpretability, as not all feature spaces within the hidden layers of the network map cleanly to natural language.

Current methods that explain the decision making processes of transformer-stack architectures focus on the embedding layer. However, these methods often result in confusing or redundant explanations, as information gets muddled passing through multiple layers of transformers in the stack. Along with (Rogers et al., 2020), we hypothesize that a more meaningful, clear, decision-oriented representation exists in solely the later layers of the network. In this paper, we propose a method that only captures the signal of the later layers of the transformer stack and projects it back onto the token space of natural language. Our method can be paired with any layer-based saliency metric, explicitly accounts for multiple layers of self-attention mechanisms and reflects the implications of the complex pre-training and task specific fine-tuning on the layers in the architecture. We validate this method both with objective measures of model importance (Hiding / Revealing Game) and human measures of external consistency (Token Overlap), and demonstrate significant improvements over the state of the art.

Our contributions are:

1. We propose a computationally-efficient method using a pre-trained language model (LM) head to "decode" the hidden layers by mapping their features back onto the token space and present its application to an example saliency approach (Grad-CAM (Selvaraju et al., 2017)) for both binary and multi-class classification problems. Our method not only requires no additional training or labels and works for any saliency method that calculates layer-specific saliency, but prioritizes task-specific information over language structure.

2. We demonstrate improvement over state of the art methods for natural language explainability in two objective measures. In the *Hiding / Revealing Game*,

[1] STR, 600 West Cummings Park, Woburn, MA 01801, USA. Correspondence to: Elizabeth M. Hou <elizabeth.hou@str.us>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

we show that the removal / addition of tokens we believe are important and damages / improves the performance of a network more than our competitors. In *Token Overlap*, we show that our method has dramatically fewer tokens that are important for multiple classes; indicating that our important tokens are truly indicative of the class in question. Notably, we achieve these improvements despite not directly optimizing for either metric, suggesting a robust result.

2. Related Work

The concept of feature attribution or saliency scores initially began in computer vision (CV) where the interest lay in being able to explain the object detection and classification decisions of convolutional neural networks (CNNs). A large swath of literature has arisen in the CV domain on model explainers that broadly fall into a several overlapping categories: gradient-based methods, propagation-based methods, and occlusion-based methods (Simonyan et al., 2014; Zeiler & Fergus, 2014; Springenberg et al., 2014; Bach et al., 2015; Noh et al., 2015; Zhou et al., 2016; Selvaraju et al., 2017; Sundararajan et al., 2017; Shrikumar et al., 2017; Lundberg & Lee, 2017; Smilkov et al., 2017; Fong & Vedaldi, 2017; Zhang et al., 2018; Gu et al., 2018; Omeiza et al., 2019); although some avoid these categories (Castro et al., 2009; Ribeiro et al., 2016).

The natural-language processing (NLP) community has adopted, extended, and introduced new variants of these methods for both simpler long short-term memory (LSTM) architectures (Li et al., 2016; Arras et al., 2017; Kádár et al., 2017) and more complex state-of-the-art transformer-stack-based architectures (Guan et al., 2019; Wallace et al., 2019; De Cao et al., 2020; Chefer et al., 2021b; Hase et al., 2021; Feldhus et al., 2021). Additionally, with the language domain and their attention-based architectures came another category of explainability methods that either visualize or use attention weight values for explanations (Bahdanau et al., 2014; Martins & Astudillo, 2016; Strobelt et al., 2018; Liu et al., 2018; Thorne et al., 2019; Kobayashi et al., 2020; Hao et al., 2021). However, this new category did not come without its share of controversy, with many papers questioning or defending their explanatory power (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019; Serrano & Smith, 2019; Pruthi et al., 2019; Vashishth et al., 2019; Bastings & Filippova, 2020).

The transfer of explainability techniques from CNN and LSTM architectures to far larger and more complex architectures with stacks of multi-headed self-attention mechanisms has proven challenging. Unlike CNNs and LSTMs, the transformer stack has little cognitive motivation, instead relying on a pre-training regime over a massive corpora to learn language structure. Many works have attempted to derive meaning from the learned structures in the architecture including (Voita et al., 2019; Michel et al., 2019) who study the role of multiple heads, other approaches (mentioned previously) study information flow in the self-attention mechanism as part of their attention based explainers, and still others simply probe and visualize the overall architecture (Tenney et al., 2019; Kovaleva et al., 2019; Vig, 2019; Jawahar et al., 2019; Rogers et al., 2020; Likhosherstov et al., 2021). However, despite these works, explainability methods for transformer-stack architectures fail to account for a principal component of the architecture, the *stacking of transformer blocks*. Unlike previous works, we propose a saliency method that explicitly accounts for its representation after multiple layers of self-attention mechanisms and reflects the implications of the complex pre-training and task specific fine-tuning on the layers in the architecture.

3. Proposed Saliency Method

We propose a method for producing explanations for decisions made by language models with encoder-based transformer architectures such as BERT (Devlin et al., 2018) and RoBERTa (Vaswani et al., 2017). We extend saliency techniques from the CV domain to the NLP domain in a way that tracks the provenance of intermediate layers in the original input space. This allows our method to capture layer specific information about the contributions of each token in an input sequence to a language model's final decision.

In the following subsections, we use Grad-CAM as the driving example for a layer-wise saliency technique. However, as will be seen in Algorithm 1, our method is agnostic to the specific saliency method as long as the method can calculate layer specific scores. We show that by using a language model head, we can compute interpretable explanations at any layer and for any saliency metric. The novelty of our overall method is the ability to assign explanatory power to the original tokens in the input sequence from scores calculated using information only downstream from a specific hidden layer in a transformer stack.

3.1. Calculating Layer Saliency Scores

One way to calculate scores that explain the contributions of a token is using Gradient-weighted Class Activation Mapping (Grad-CAM), a gradient-based saliency method which was first proposed in (Selvaraju et al., 2017) to produce visual explanations for CV problems. Grad-CAM is one of several gradient-based methods that has the advantage of being able to perform class-discriminative localization for any CNN-based models without requiring architectural changes or re-training.

We compute the gradient of the predicted score y^c (before softmax) for class c with respect to an output of a transformer block h_l. These gradients can be viewed as weights that capture the "importance" (for a specific class c) of each of the K features for each element of the output sequence. Specifically for Grad-CAM, this represents a partial linearization of the model downstream from h_l. We calculate the Grad-CAM scores as a weighted combination of the features in the output and use a ReLU function to assign no importance to elements with negative scores. Explicitly, let

$$\alpha_l^c = \left[\dots, \frac{\partial y^c}{\partial h_l^k} h_l^k, \dots\right] \qquad \forall k = 1, \dots, K \tag{1}$$

$$s_l^c = g(\alpha_l^c) = ReLU(\sum_{k=1}^{K} \alpha_l^{c,k}) \tag{2}$$

where the h_l^k are the column vectors corresponding to the K features of the output of layer l in the stack of transformer blocks, α_l^c is a matrix of size $n \times K$ whose elements are the scores for each of the features of the output h_l, and s_l^c is vector of size n whose elements are the scores aggregated over all features.

Note, the scores s_l^c can be replaced with any method[1] that calculates layer specific scores for each of the features by replacing (1) with another matrix of layer specific feature scores and some other feature aggregation function $g(\cdot)$.

If we calculate the scores with respect to 0-th layer outputs h_0 (embeddings of token, segment, and positional information), the elements of the score vector correspond directly to the tokens of the input sequence. However, once an input sequence passes through a transformer block (described explicitly in (5) in Section 3.2), this relationship no longer holds. Unlike CNNs where there is a clear provenance between the pixels and the outputs of the convolution filters, the multi-headed self-attention mechanism of transformers are far more complicated. The receptive fields of a CNN are local patches, whereas the receptive fields of the outputs of a transformer block are far more global consisting of the entire input. This is because each self-attention head uses the entire input to learn new representations for some subset of the features making each element of the output a function of all elements of the input. Many works (Vig, 2019; Tsai et al., 2019; Likhosherstov et al., 2021) have attempted to attribute meaning to the attention mechanism with varying levels of success; however, they primarily focus on a single transformer block. The meaning of an output sequence when the input sequence is passed through multiple transformer blocks in a stack is even less clear.

Thus, while it is relatively easy to calculate scores s_l^c with respect to the embeddings ($l = 0$), which already lie in the token space, it may not necessarily produce explanations that are most relevant to a models prediction. (Rogers et al., 2020) surveys 150 papers and derives potential explanations for the roles of the layers of the BERT model. They conclude that the lower layers have the most information about linear word order (i.e. the linear position of a word in a sentence (Lin et al., 2019)), the middle layers contain syntactic information, and the final layers are the most task-specific (Jawahar et al., 2019). Therefore, it would be worthwhile to also explore the explanatory power of the saliency scores of the other layers $l > 0$ where only information in the network downstream from that layer is included in the score. By only capturing information downstream from a specific layer, we ignore potentially task-irrelevant information in the earlier layers of the network.

3.2. Interpreting the Hidden Layers

In order to calculate saliency scores, e.g. (2), that only capture information downstream from a specific layer, we need to project these scores into a space where the elements of the projected vector correspond directly to the original tokens of an input sequence. This allows us to have a meaningful version of the scores where we can directly understand the contributions of each token. And because the elements of a scores vector correspond directly to the elements of a transformer block's outputs, the problem of projecting scores into a token space is equivalent to the problem of projecting outputs into a token space. Thus this can be mathematically formulated as finding a mapping f that minimizes the loss $\mathcal{L}(\cdot)$ between the output of a transformer block and its closest possible token space representation t,

$$\arg\min_f \mathcal{L}(t, f(h_l)) \tag{3}$$

where f holds for any layer l in a transformer stack and t is a $n \times V$ right stochastic matrix whose rows lie in the token space $\mathcal{T}$ defined as the surface of a V-dimensional unit hypersphere. The axes of this hypersphere correspond to the V tokens in a vocabulary, so any point on this hypersphere's surface is the weighted contributions of the tokens.

One of the pre-training tasks of models with encoder-based transformer architectures (e.g. BERT) is the masked language model task, which is trained to minimize exactly this loss when $\mathcal{L}(\cdot)$ is a cross-entropy function, t is the original input sequence, and the layer $l = L$ is the final transformer block in the stack. The masked language model task trains two functions: $f^{base}(\cdot)$ that represents the transformer-stack and $f^{lm}(\cdot)$ the language model (LM) head that minimizes $\mathcal{L}(t^i, f^{lm}(h_L^i))$ where $h_L^i = f^{base}(t^i)$ is the i-th element of the output of the full transformer stack and t^i is a one-hot vector representing the token at the i-th element of the original input sequence.

While we now have a function $f^{lm}(\cdot)$ that solves a specific version of (3), we still need to understand the function's

[1] See Appendix (A) for examples with other methods.

role in encoder-based transformer architectures. The LM head takes in a row i of the $n \times K$ final transformer-stack output and transforms it to lie in the same space as the corresponding row of the $n \times V$ one-hot matrix of the original input sequence. It decomposes as

$$f^{lm}(h_L^i) = \sum_{j=1}^{V} \hat{P}_L^{ij} \boldsymbol{e}^{(j)} \tag{4}$$

where $\boldsymbol{e}^{(j)}$ is a $1 \times V$ basis vector with a one in column j and a zero elsewhere representing j-th dimension of the token space $\mathcal{T}$ and $\hat{P}_L$ is a $n \times V$ right stochastic matrix with each row i containing the (after softmax) prediction probabilities of being the j-th token in the vocabulary. Thus, because the basis vector $\boldsymbol{e}^{(j)}$s correspond to tokens where j is the token's position in the vocabulary, we can interpret $\hat{P}_L^{ij}$ as the amount of influence the j-th token has on i-th element of h_L.

However, the masked language model task is training a function specifically for the final output of the transformer stack h_L. In order to extrapolate the effects of the $f^{lm}(\cdot)$ function to the output of any layer l, we must understand the most complicated part of an encoder-based transformer architecture, the self-attention mechanism. As previously studied in (Likhosherstov et al., 2021; Tsai et al., 2019), the output of the self-attention mechanism can be expressed as

$$X' = AXW_V \tag{5}$$

where $A = softmax(\frac{XW_Q W_K^T X^T}{\sqrt{d}})$ is the normalized self-attention matrix, W_Q, W_K, W_V are the query, key, and value weight matrices, and d is hidden dimension of the self-attention mechanism. The self-attention matrix A is a weighted similarity or kernel gram matrix between the elements of the input X, and the features of the output X' are weighted combinations of the features of the inputs. The multi-headed mechanism simply combines various self-attention mechanisms in a weighted fashion. The rest of the transformer block consists of a feed-forward component and some layer additions and normalizations. Thus we can describe outputs of a transformer block overall as approximately a weighted combination of its inputs. Similarly, stacking transformer blocks together results in further weighted combinations of the original input sequence.

This leads to the key idea that because the outputs of any stack of transformer blocks are a weighted combination of original K features, they lie in the same continuous feature space $\mathbb{R}^K$. Unlike the original token space $\mathcal{T}$, this feature space does not have an easily interpretable meaning. We conjecture that because the pre-training tasks are performed over an enormous corpus (Wikipedia etc.), the learned function $f^{lm}(\cdot)$ is estimating the map between $\mathbb{R}^K$ and $\mathcal{T}$ where the feature space is much smaller than the token vocabulary space $K << V$. Thus the $f^{lm}(\cdot)$ function acts as a universal decoder that predicts the likeliest combination of tokens, i.e. basis vectors $\boldsymbol{e}^{(j)}$ of $\mathcal{T}$, that make up $f^{lm}(h_l)$ where $\hat{P}_l$ are the prediction probabilities. We provide a simple example illustrating this process in Figure 1 to provide geometric intuition.

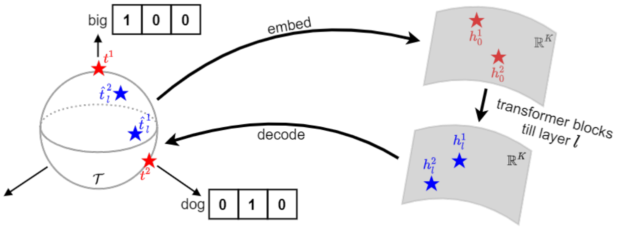


Figure 1. The input sequence "big dog" is tokenized into two tokens t^1 and t^2 that lie on the unit hypersphere $\mathcal{T}$ and are then embedded into having K continuous features (h_0^1 and h_0^2). A series of l transformer blocks is applied to the embedded input sequence to produce h_l^1 and h_l^2, which are decoded back onto $\mathcal{T}$ with the $f^{lm}(\cdot)$ function. The outputs ($\hat{t}_l^1$ and $\hat{t}_l^2$) of the transformer blocks in $\mathcal{T}$ can be interpreted as weighted combinations of the original tokens t^1 and t^2.

While the rows of $\hat{P}_l$ can be interpreted as the likelihood that each element of h_l is a certain token in the vocabulary, the columns of $\hat{P}_l$ can analogously be interpreted as the amount of influence each token has on h_l. However, because we are only interested in the contributions of tokens from the input sequence, we can subset the V columns of $\hat{P}_l$ to only the T columns that correspond to the unique input sequence tokens. Let $\hat{D}_l$ be a $T \times n$ matrix where the rows of $\hat{D}_l$ are the columns of $\hat{P}_l$ that correspond to the tokens in the input sequence and the columns indices of $\hat{D}_l$ correspond to the elements of h_l. Now that we have a way to account for the contributions of tokens in an input sequence to a hidden layer, we can calculate a layer's saliency scores with respect to these tokens as

$$s_l^c = ReLU(\sum_{k=1}^{K} \hat{D}_l \alpha_l^{c,k}) \tag{6}$$

where the elements of s_l^c are weighted combinations of the features scores for layer l from (1) with each element being a different weight according to the rows of $\hat{D}_l$. This method can also be viewed as a generalization of (2) where the feature aggregation function $g(\cdot)$ has been modified to include additional weights.

These saliency scores s_l^c capture the importance of each token in the input sequence to the models decision using *only* information in the model that is downstream from a specific layer l in a transformer stack. Thus we can view the layer choice l as a control for the amount of model information used in a saliency score.

Additionally if we only want to allow contributions from the most important tokens, we can restrict each $\hat{s}_l^i$ in (6) to be a weighted combination of only the output scores where the input token t_i is an top ranked contributor. We show pseudocode for our approach in Algorithm 1 which takes as inputs a tokenized input sequence t, a layer choice l, a threshold τ for the number of contributions from top ranked tokens, a feature aggregation function for a saliency method $g(\cdot)$, a LM task head from a pre-trained model $f^{lm}(\cdot)$, and a fine-tuned classification model $m(\cdot)$ where $m_l^{base}(\cdot)$ is the output of layer l in the transformer stack and $m^{class}(\cdot)$ is the classification task head. For a given layer l, estimate the features scores $\hat{\alpha}_l$ for the most likely prediction $\hat{y}$ and the output of the l-th block in the transformer stack h_l (Lines 1-3). Then estimate the token contributions $\hat{D}_l$ as a subset of $\hat{P}_l$, the LM probability predictions for h_l onto the input token sequence t (Lines 4-5). Finally the saliency scores $\hat{s}_l$ are a weighted sum of output scores $\hat{\alpha}_l$ where the non-zero weights are the top τ ranked values in the columns of $\hat{D}_l$ (Lines 6-12). These scores are then aggregated with a feature aggregation function $g(\cdot)$, for Grad-CAM $g(\cdot) = ReLU(\sum_{k=1}^{K} \cdot)$.

Algorithm 1 Transformer-stack architectures embed a discrete vocabulary into a lower dimensional continuous space where layers in the stack merely transform it within this space. Our approach generates a decoder from the low-dim latent space back to the original token space.

```
**Input:** $t, l, \tau, g(\cdot), f^{mlm}(\cdot), m(\cdot)$
 1: $h_l = m_l^{base}(t)$
 2: $\hat{y} = \max m^{class}(m^{base}(t))$
 3: $\hat{\alpha}_l = \frac{\partial \hat{y}}{\partial h_l^k} h_l^k$ or some other feature score for layer $l$
 4: $\hat{P}_l = f^{lm}(h_l)$
 5: $\hat{D}_l^i = (\hat{P}_l^j)^\top \quad \forall j$ corresponding to tokens $t^i$
 6: for $i = 1$ to $T$ do
 7:   for $j = 1$ to $n$ do
 8:     if $\hat{D}_l^{ij}$ in top $\tau$ ranked values of $\hat{D}_l^j$ then
 9:       $\hat{s}_l^j \mathrel{+}= g(\hat{D}_l^{ij} \hat{\alpha}_l^j)$
10:     end if
11:   end for
12: end for
**Output:** $\hat{s}_l$
```

While the weights for the classification model $m(\cdot)$ change when fine-tuned to a specific dataset, they are still initialized at the pre-trained values. In practice because the weights of the model base m_l^{base} change very little during fine-tuning, and in some training schemes are restricted not to change, we can still assume the transformer block outputs h_l will still lie in $\mathbf{R}^K$. Thus because $f^{lm}(\cdot)$ estimates the map from $\mathbf{R}^K$ to $\mathcal{T}$, it is still able to decode h_l despite being the outputs of a model with different weights.

4. Experiments

In this section, we detail our experimental results on two benchmark classification task datasets[2]: SST-2 (Socher et al., 2013) a binary classification dataset that is one of the the General Language Understanding Evaluation (GLUE) (Wang et al., 2019) tasks and AG News (Zhang et al., 2015) a subset (4 largest classes) of news articles from more than 2,000 news sources gathered by (Gulli, 2005). We implemented our approach (labelled Decoded Grad-CAM) on a RoBERTa base from HuggingFace (Wolf et al., 2020) using Grad-CAM for the saliency scores[3] and compared against numerous other explainability methods that have been trained and provided by the AllenNLP Interpret (Wallace et al., 2019) and ThermoStat (Feldhus et al., 2021) Python packages. To the best of our abilities, we have attempted to mimic the training regimes described by their respective packages for all competing models' explainability methods. However, in order to maintain consistency across all experiments and improve visibility, we have chosen to always use the standard RoBERTa base model with a 12 layer transformer stack. For further details on the experimental setup see Section A.1.

4.1. The Hiding /Revealing Game

In order to evaluate the explainability of a token, we use the Hiding Game (Fong & Vedaldi, 2017; Castanon & Byrne, 2018) and an inverse variant of it, which we will call the Revealing Game. For NLP, the Hiding Game iteratively obscures the least important tokens according to some score attributed with the token, replaces them with a [MASK] token, and removes them from the self-attention mechanism. The Revealing Game does the opposite and starts with a completely masked sequence and iteratively reveals the most important tokens according to their score. For both games, the prediction accuracy is periodically calculated at percentages of the total sequence length (ignoring [PAD] tokens). Similar variants such as positive / negative perturbations (Chefer et al., 2021a) or using masking in (Hase et al., 2021) have also been used for evaluating the explainability of methods. In addition to the AllenNLP Interpret and ThermoStat explainers, we also compare against a random baseline that is averaged over 20 random perturbations.

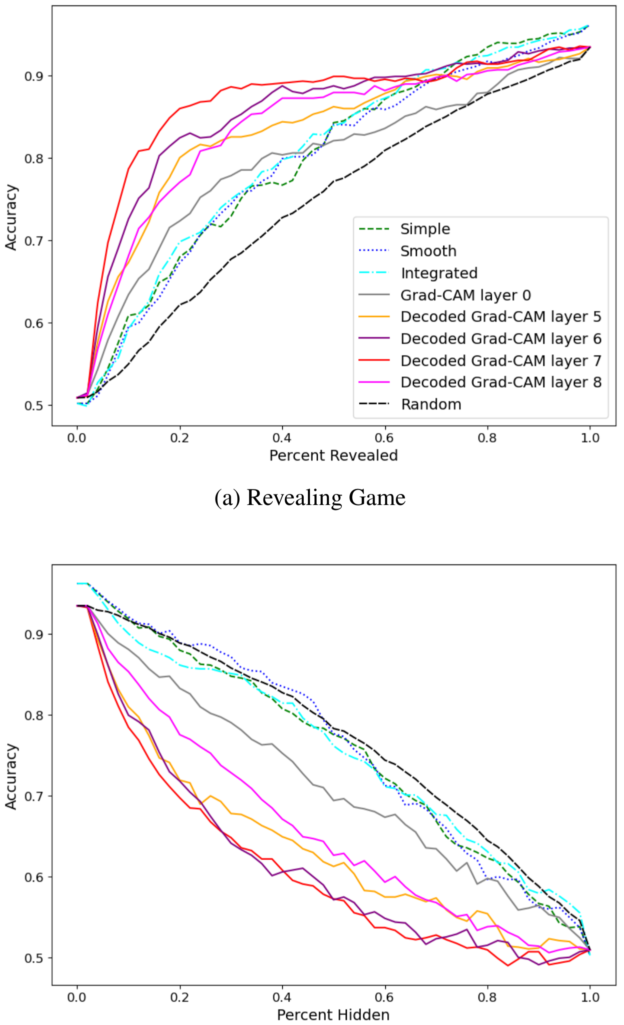
In Figure 2, we use the Hiding / Revealing Game to evaluate the accuracy of various explainability methods on the SST-2 sentiment classification dataset. We show the performance of the best layers[4] of our Decoded Grad-CAM (for visibility) against AllenNLP Interprets implementation of

[2] For a full description of the datasets, see Section A.1.

[3] See Appendix (A.2.1) for additional examples using the method from (Simonyan et al., 2014) for saliency scores.

[4] See Section A.2 for all layers

the Simple, Smooth, and Integrated methods along with a layer 0 (vanilla) version of Grad-CAM.



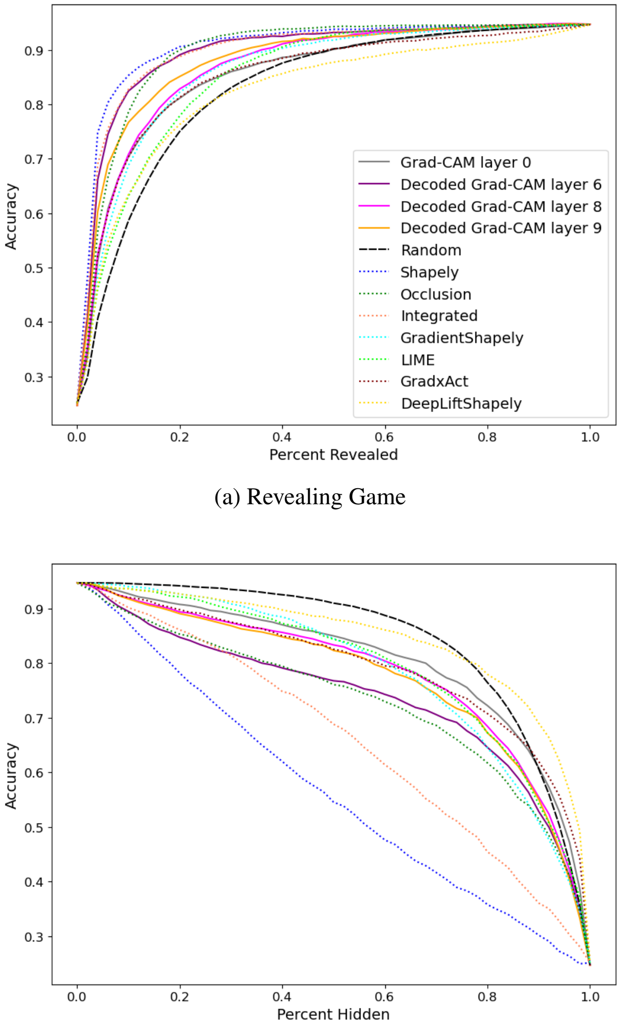
(a) Revealing Game

(b) Hiding Game

Figure 2. Our Decoded Grad-CAM method against vanilla Grad-CAM, AllenNLP Interpret explainers, and a random baseline on the SST-2 binary sentiment classification dataset.

For the Revealing Game, the accuracy of our layers shoots up very quickly after the first couple of tokens are revealed, implying that those tokens are very important to the model's classification decision. For the Hiding Game, all four of our layers have steeper drops in accuracy, which implies that the tokens being masked are more important as they dramatically affect the accuracy. Note that we are using a RoBERTa-base model, which is smaller than the RoBERTa-large model used by the AllenNLP Interprets explainers and is the reason for the small gap in accuracy for the full input sequence. Despite having a smaller underlying model, our Decoded Grad-CAM at layer 7 outperforms the explainers on a larger model up until the vast majority ($\approx 70\%$) of important tokens have been revealed.

We also apply the Hiding / Revealing Game to the AG News dataset, which is a multi-class topic classification task, and evaluate against numerous ThermoStat explainers. We show the accuracy of the best layers[5] against all the ThermoStat explainers in Figure 3.



(a) Revealing Game

(b) Hiding Game

Figure 3. Our Decoded Grad-CAM method against vanilla Grad-CAM, ThermoStat explainers, and a random baseline on the AG News four topic classification dataset.

Many of the ThermoStat explainers are specifically built to probe for changes in predictions from changes in tokens and are essentially optimized to do well in the Hiding Game. However due to this perturbation construction, many of these methods (Integrated, LIME, Occlusion, and Shapely) are also extremely computationally ex-

[5] See Section A.2 for all layers

pensive requiring many passes forward through the model and the ThermoStat package was specifically constructed to improve accessibility (at least for benchmark datasets) to these explainers (Feldhus et al., 2021). In contrast, our decoded layer saliency method is applicable to any trained model without requiring any re-training and only requires one backward pass, which is far more computationally efficient. Additionally, because our method is not probing for changes from a correct class to an incorrect one, it does not require labels. This makes it useful as an explainer even in scenarios where a user only has access to a trained model and does not have access to any training or labelled data.

Despite this, our Decoded Grad-CAM layer 6 outperforms the majority of the ThermoStat explainers including the computationally heavy LIME method. Similar to the results for the SST-2 dataset, we see the accuracy of our best layers shoot up quickly in the Revealing Game with layer 6 having very competitive results with ThermoStats' Shapely, Occlusion, and Integrated methods. For the Hiding Game, our layers do not exhibit as dramatic of a drop, but still do significantly better than most ThermoStat explainers and our layer 6 is competitive with the all except the Shapely method until $\approx 40\%$ of tokens are hidden.

We also calculate the Area Under the Curve (AUC) for the explainers in both figures above in Table 1, where the best layer explainer of our method is **bolded** and the best competing explainer is ***italicized***. From Figure 3's AUCs, for the Revealing Game, the difference in performance between the top four best explainers is extremely minor with only 0.011 gap between the first and fourth place methods. For the Hiding Game, the Shapely explainer is clearly the best; however our layer 6 still has respectable performance being only 0.008 worse than the Occlusion method and 0.078 worse than the Integrated method.

Another noteworthy observation is that our best performing layers (5-8 for the SST-2 dataset and 6, 8 and 9 for the AG News dataset) roughly correspond to the "middle layers" described by (Rogers et al., 2020). Thus our saliency method is only including information downstream from these layers, namely those corresponding to the "later layers", which are described to be more task-specific. These conclusion are also supported by experiments in (Jawahar et al., 2019) where they conclude that the "later layers" of the model are capture information about semantic tasks. Semantic tasks (e.g. randomly replacing nouns/verbs or swapping clausal conjuncts) often operate on parts of text that are also the most useful for classification; thus we are in agreeance about the role of these "later layers".

4.2. Token Overlap

While the previous experiments are a good way to evaluate the affect of tokens on a model's decision making, they don't actually provide any indication of explainability to a human. In order to judge the human intuitiveness of the explanations, we should also consider the actual meanings of the top ranking tokens. Thus, we aggregate the scores of all tokens for all input sequences in a predicted class and weight their total score by how rarely they occur in everyday language i.e. the inverse document frequency of a random collection of 50,000 Wikipedia articles. The intuition behind this is that tokens that have high importance scores *and* occur often in the input sequence of a predicted class relative to usage in common language are representative of that class. By aggregating over all input sequences of a predicted class, we also reduce the rewarding of one-off tokens that only explain the model's decisions for that specific input sequence. We can visualize the most important tokens for each predicted class in word clouds shown in Figures 9 and 11 in Section A.3.

Table 1. Area Under the Curve for Revealing Game (↑ is better) and Hiding Game (↓ is better)

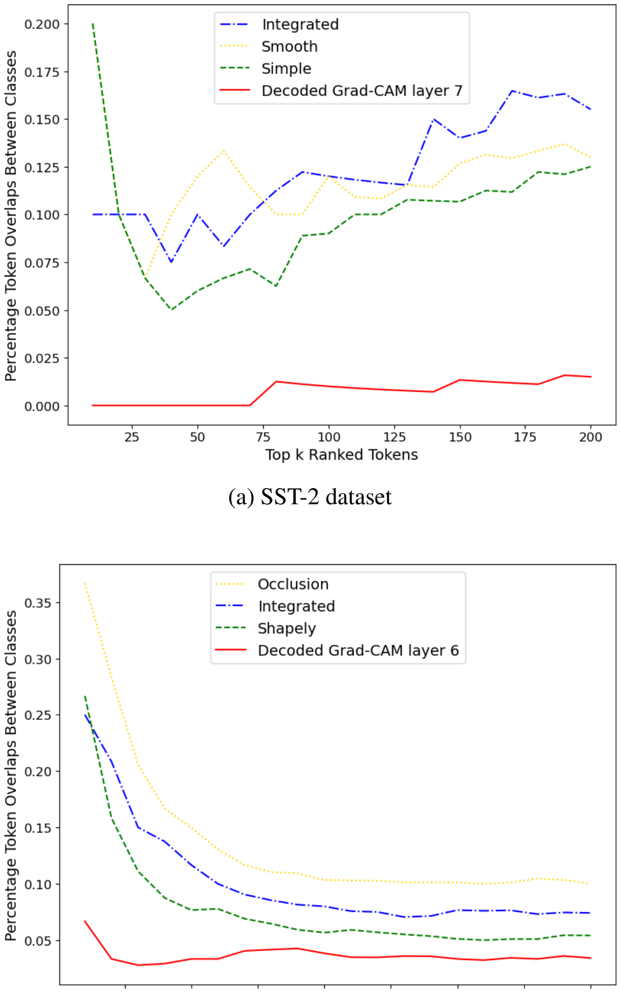
Explainer	Revealing	Hiding
Grad-CAM l_0	0.797	***0.713***
Decoded Grad-CAM l_5	0.832	0.64
Decoded Grad-CAM l_6	0.854	0.617
Decoded Grad-CAM l_7	**0.867**	**0.609**
Decoded Grad-CAM l_8	0.836	0.66
Simple	0.799	0.756
Smooth	0.795	0.762
Integrated	***0.804***	0.756
Random	0.748	0.77

(a) SST-2 Dataset (Figure 2)

Explainer	Revealing	Hiding
Grad-CAM l_0	0.853	0.804
Decoded Grad-CAM l_6	**0.895**	**0.739**
Decoded Grad-CAM l_8	0.867	0.782
Decoded Grad-CAM l_9	0.879	0.774
Shapely	***0.905***	***0.567***
Occlusion	0.894	0.731
Integrated	0.897	0.661
GradientShapely	0.86	0.784
LIME	0.851	0.792
GradxAct	0.851	0.794
DeepLiftShapely	0.823	0.842
Random	0.829	0.841

(b) AG News Dataset (Figure 3)

Additionally, for classification tasks, tokens should disambiguate classes. So tokens that are important to a predicted class should be *indicative*, i.e. unique to a class, especially if the classes are complements of each other. For example, tokens that are strong indicators that a movie review is positive should not also be strong indicators that a movie review is negative. Therefore, we can also evaluate the representativeness of tokens deemed important to a predicted class by considering how often they appear in multiple classes. Explicitly, we count the number of top k ranked tokens that appear in every pair of classes and divide by the total count in order to get the percentage of token overlap.



(a) SST-2 dataset

(b) AG News dataset

Figure 4. Percentage of tokens that appear in multiple classes for the top k most important tokens.

We show this percentage as a function of the top k ranked tokens in Figure 4 for the best Decoded Grad-CAM layer according the Hiding / Revealing Game against the AllenNLP Interpret explainers on the SST-2 dataset and against the best ThermoStat explainers on the AG News dataset. Additionally we show, in tables in Section A.4, the actual tokens in the top 50 that appear in multiple classes of the SST-2 and AG News datasets respectively, along with the raw counts of token overlap in Figure 12.

For both datasets, our best Decoded Grad-CAM layer significantly outperforms the competing methods with very few important tokens belonging to multiple classes. Unlike the other explainability methods, our approach only incorporates information in the network that is downstream from a specific layer. Thus its does not include language structure information such as the word order or syntactic information from earlier layers that would add noise to the explainer. We can interpret from the plots that because the competing methods have many more tokens that are salient for multiple classes, these tokens may be structurally important, but not class discriminate. The removal of these structurally important tokens may also be causing an out of distribution effect in the Hiding Game (Hase et al., 2021) and biasing their good performance. We also provide some example snippets of input sequences highlighted by the methods in the above figures in tables in Section A.5. These examples provide an additional way for a human to directly visualize and interpret the explainability of the methods for a particular input sequence.

5. Discussion

In this paper, we have presented an approach for measuring the importance of tokens to a classification task based on the information encoded in the hidden layers of the transformer stack. Consistent with previous research into the meanings of these intermediate layers in large-scale language models, we explicitly confirm, through multiple experiments, that the later layers generate better task-specific human explainability. Our approach works with any score-generation method that generates layer-specific importance scores and requires no re-training. Most importantly, it shows that information in the later layers of the transformer stack are more important for model classification performance (The Hiding Game) as well as for human consistency (Token Overlap). In the future, we look to extend this work to tasks beyond classification. We also plan to further explore and leverage the geometric relationship between the feature embedding and token spaces first established in this paper.

Acknowledgments

This material is based upon work supported by the United States Air Force under Contract No. FA8650-19-C-6038 and FA8650-21-C-1168. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Air Force or Department of Defense.

References

Arras, L., Montavon, G., Müller, K.-R., and Samek, W. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206*, 2017.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bastings, J. and Filippova, K. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? *arXiv preprint arXiv:2010.05607*, 2020.

Castanon, G. and Byrne, J. Visualizing and quantifying discriminative features for face recognition. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 16–23. IEEE, 2018.

Castro, J., Gómez, D., and Tejada, J. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730, 2009.

Chefer, H., Gur, S., and Wolf, L. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 397–406, 2021a.

Chefer, H., Gur, S., and Wolf, L. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 782–791, 2021b.

De Cao, N., Schlichtkrull, M., Aziz, W., and Titov, I. How do decisions emerge across layers in neural models? interpretation with differentiable masking. *arXiv preprint arXiv:2004.14992*, 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Feldhus, N., Schwarzenberg, R., and Möller, S. Thermostat: A large collection of nlp model explanations and analysis tools. In Adel, H. and Shi, S. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2021.

Fong, R. C. and Vedaldi, A. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pp. 3429–3437, 2017.

Gu, J., Yang, Y., and Tresp, V. Understanding individual decisions of cnns via contrastive backpropagation. In *Asian Conference on Computer Vision*, pp. 119–134. Springer, 2018.

Guan, C., Wang, X., Zhang, Q., Chen, R., He, D., and Xie, X. Towards a deep and unified understanding of deep neural models in nlp. In *International conference on machine learning*, pp. 2454–2463. PMLR, 2019.

Gulli, A. Ag's corpus of news articles, 2005. URL http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html.

Hao, Y., Dong, L., Wei, F., and Xu, K. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 12963–12971, 2021.

Hase, P., Xie, H., and Bansal, M. The out-of-distribution problem in explainability and search methods for feature importance explanations. *Advances in Neural Information Processing Systems*, 34:3650–3666, 2021.

Jain, S. and Wallace, B. C. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

Jawahar, G., Sagot, B., and Seddah, D. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Kádár, A., Chrupała, G., and Alishahi, A. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780, 2017.

Kobayashi, G., Kuribayashi, T., Yokoi, S., and Inui, K. Attention is not only a weight: Analyzing transformers with vector norms. *arXiv preprint arXiv:2004.10102*, 2020.

Kovaleva, O., Romanov, A., Rogers, A., and Rumshisky, A. 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)*, 2019.

Li, J., Monroe, W., and Jurafsky, D. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

Likhosherstov, V., Choromanski, K., and Weller, A. On the expressive power of self-attention matrices. *arXiv preprint arXiv:2106.03764*, 2021.

Lin, Y., Tan, Y. C., and Frank, R. Open sesame: Getting inside BERT's linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 241–253. Association for Computational Linguistics, August 2019. doi: 10.18653/v1/W19-4825. URL https://aclanthology.org/W19-4825.

Liu, S., Li, T., Li, Z., Srikumar, V., Pascucci, V., and Bremer, P.-T. Visual interrogation of attention-based models for natural language inference and machine comprehension. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2018.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Martins, A. and Astudillo, R. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pp. 1614–1623. PMLR, 2016.

Michel, P., Levy, O., and Neubig, G. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Noh, H., Hong, S., and Han, B. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528, 2015.

Omeiza, D., Speakman, S., Cintas, C., and Weldermariam, K. Smooth grad-cam++: An enhanced inference level visualization technique for deep convolutional neural network models. *arXiv preprint arXiv:1908.01224*, 2019.

Pruthi, D., Gupta, M., Dhingra, B., Neubig, G., and Lipton, Z. C. Learning to deceive with attention-based explanations. *arXiv preprint arXiv:1909.07913*, 2019.

Ribeiro, M. T., Singh, S., and Guestrin, C. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Rogers, A., Kovaleva, O., and Rumshisky, A. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Serrano, S. and Smith, N. A. Is attention interpretable? *arXiv preprint arXiv:1906.03731*, 2019.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In *International conference on machine learning*, pp. 3145–3153. PMLR, 2017.

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

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. Smoothgrad: removing noise by adding noise. In *ICML Workshop on Visualization for Deep Learning*, 2017.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Strobelt, H., Gehrmann, S., Behrisch, M., Perer, A., Pfister, H., and Rush, A. M. S eq 2s eq-v is: A visual debugging tool for sequence-to-sequence models. *IEEE transactions on visualization and computer graphics*, 25(1): 353–363, 2018.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Tenney, I., Das, D., and Pavlick, E. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

Thorne, J., Vlachos, A., Christodoulopoulos, C., and Mittal, A. Generating token-level explanations for natural language inference. *arXiv preprint arXiv:1904.10717*, 2019.

Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. Transformer dissection: A unified understanding of transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

Vashishth, S., Upadhyay, S., Tomar, G. S., and Faruqui, M. Attention interpretability across nlp tasks. *arXiv preprint arXiv:1909.11218*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vig, J. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*, 2019.

Voita, E., Talbot, D., Moiseev, F., Sennrich, R., and Titov, I. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*, 2019.

Wallace, E., Tuyls, J., Wang, J., Subramanian, S., Gardner, M., and Singh, S. AllenNLP Interpret: A framework for explaining predictions of NLP models. In *Empirical Methods in Natural Language Processing*, 2019.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Wiegreffe, S. and Pinter, Y. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., and Sclaroff, S. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.

A. Appendix

A.1. Experimental Setup

Datasets:

1. SST-2 (Socher et al., 2013): This is a two class version of the Stanford sentiment analysis corpus where each sample is a full sentence from a movies review labeled as either Negative (class 0) or Positive (class 1) sentiment. It is split to have 67,349 training samples, 872 validation samples, and 1821 test samples; however the test labels are not publicly available and the validation set is commonly used for experiments in numerous paper including this one.
2. AG News (Zhang et al., 2015): This is a four class version of a corpus collected by (Gulli, 2005) from over 2000 news sources. Each sample is a full sentence from a news article labeled as belonging to the World (class 0), Sports (class 1), Business (class 2), or Sci/Tech (class 3) topics. It is split to have 120,000 training samples and 7,600 test samples.

Models:
AllenNLP Interpret on a RoBERTa large model (Wallace et al., 2019)

1. Simple (Simonyan et al., 2014): gradient of the loss with respect to each token normalized by the ℓ_1 norm
2. Smooth (Smilkov et al., 2017): average the gradient over noisy input sequences (add white noise to embeddings)
3. Integrated (Sundararajan et al., 2017): integrating the gradient with 10 samples along the path from an embedding of all zeros to the original input sequence

ThermoStat on a RoBERTa base model (Feldhus et al., 2021)

1. GRADxACT: simple element-wise product of gradient and activation
2. Integrated (Sundararajan et al., 2017): same as above, except 25 samples along the path
3. LIME (Ribeiro et al., 2016): sample 25 points around input sequence and use predictions at sample points to train a simpler interpretable model
4. Occlusion (Zeiler & Fergus, 2014): perturbation based approach, replace sliding window (3 tokens) with baseline and compute difference in prediction
5. Shapley (Castro et al., 2009): add a random permutation of tokens from the input sequence to a baseline, look at difference in prediction after each addition, perform 25 times and average over them
6. DeepLiftShap (Lundberg & Lee, 2017): approximates Shapely values, computes DeepLift attributions for each input-baseline pair, average over baselines
7. GradientShap (Lundberg & Lee, 2017): approximates Shapely values, computes the expectations of gradients by randomly sampling 5 times from the distribution of baselines

Decoded Grad-CAM layers implemented on RoBERTa base from HuggingFace (Wolf et al., 2020)

Metrics:

1. The Hiding / Revealing game: Order tokens in descending amounts of importance for each method, hiding each token one by one. Methods with a better grasp of importance will reduce the prediction accuracy of the network faster by hiding tokens that really matter. The Revealing Game is the converse, which slowly reveals important tokens. This method was used in (Fong & Vedaldi, 2017; Castanon & Byrne, 2018).
2. Percentage of Token Overlaps: For every pair of classes we count the number of tokens in the top k most-important for both classes and divide total count by $\binom{C}{2} * k$ where C is number of classes. This yields a measure of how unique the tokens we believe are important for identify a class are.

A.2. Additional Hiding / Revealing Game Results

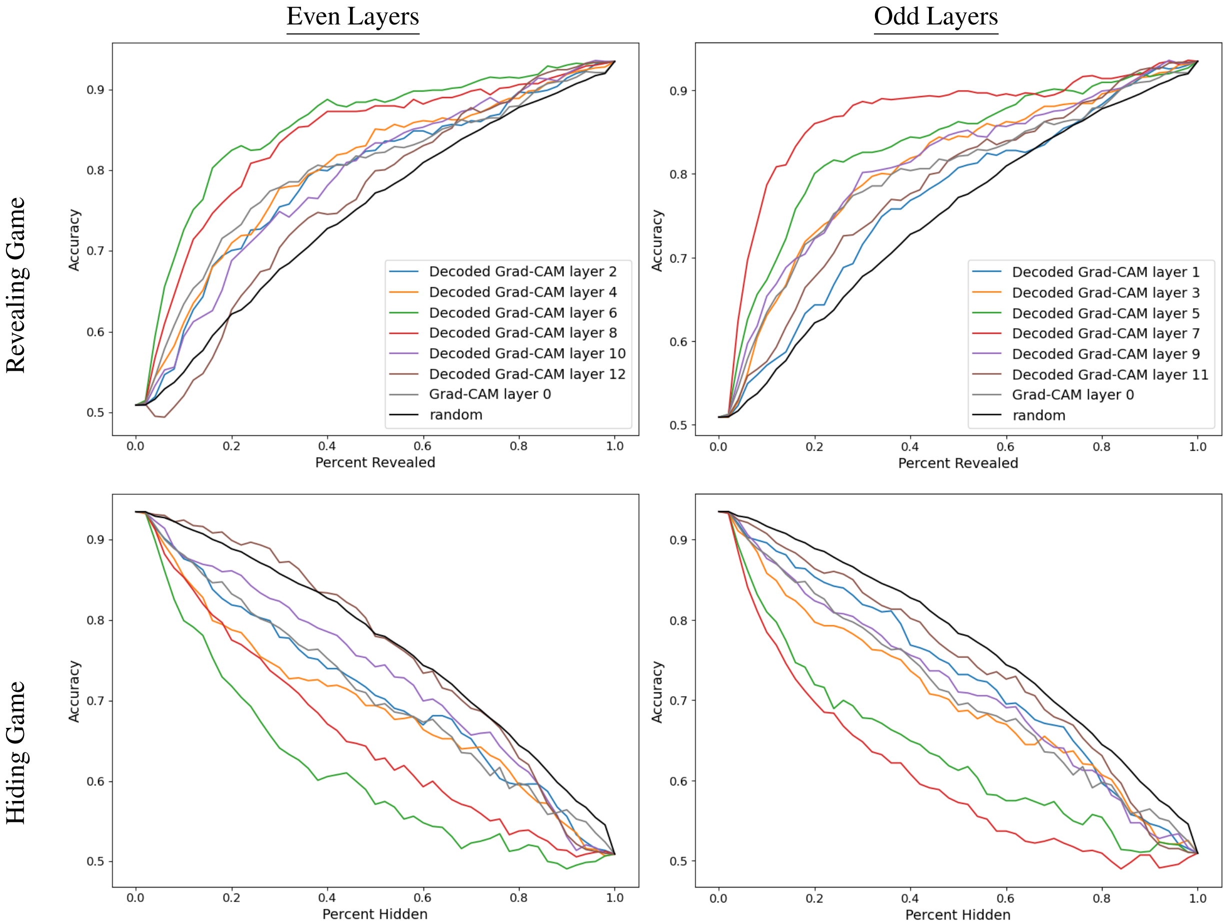


Figure 5. All Decoded Grad-CAM layers on the SST-2 dataset.

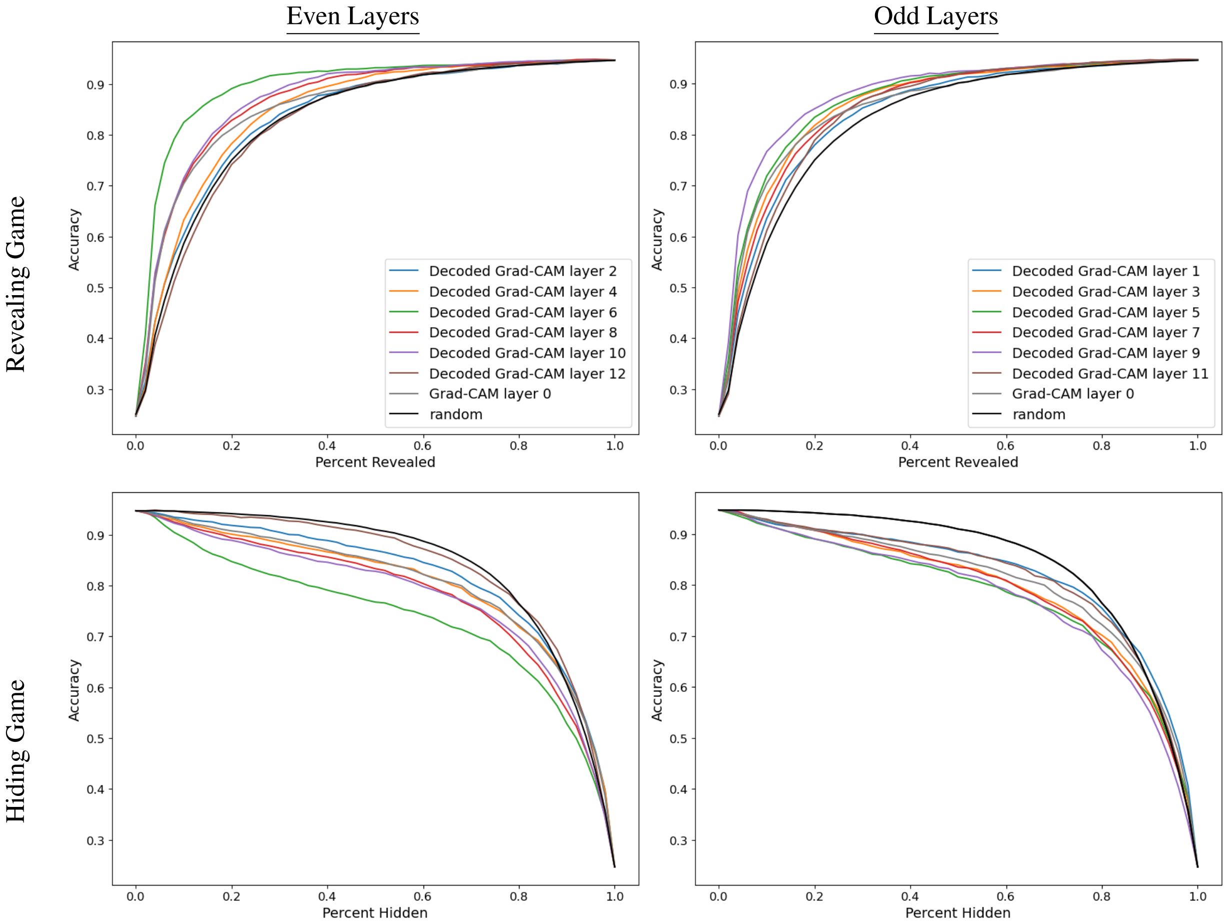
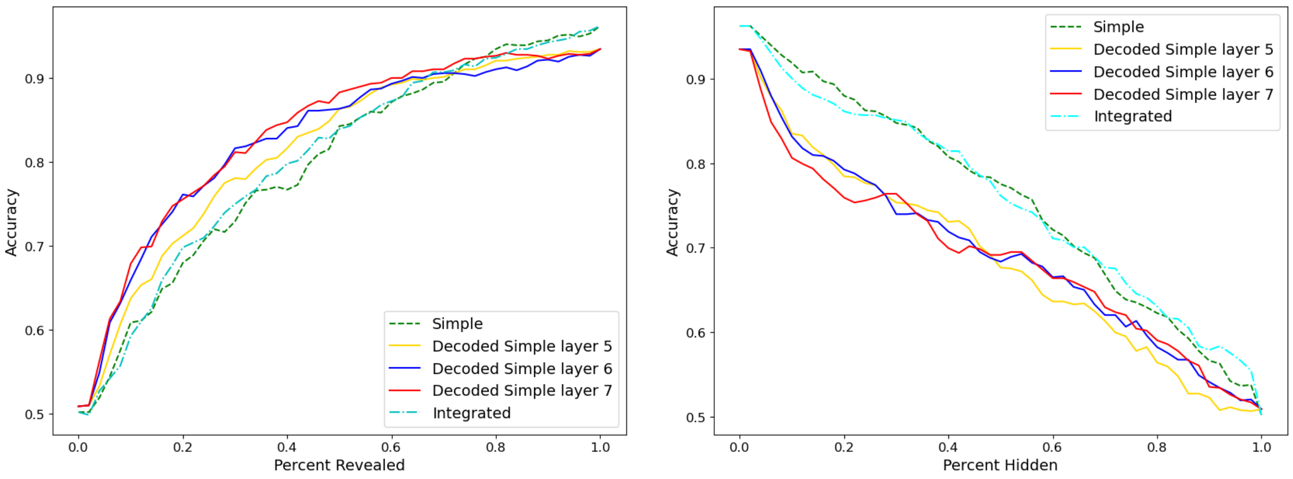


Figure 6. All Decoded Grad-CAM layers on the AG News dataset.

A.2.1. Other Saliency Score Results

In order to show generalizability of our method described in Algorithm 1 to more than Grad-CAM, we also show results replacing Grad-CAM with the "Simple" method from AllenNLP Interpret. Below we show the performance of layers 5, 6, and 7 using our decoding method versus the original "Simple" method. Note: there is a slight difference in performance due to the AllenNLP Interpret using a RoBERTa-large model and our method using a RoBERTa-base model.

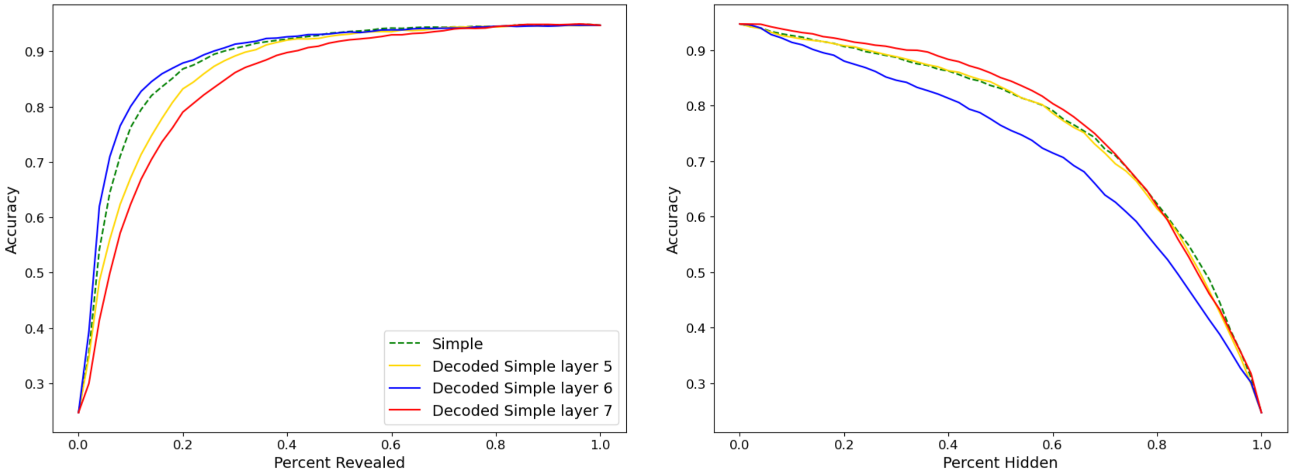
Similar to the Grad-CAM results, only including gradients above certain middle layers for the "Simple" method can have better performance than including all gradients by backpropagating to the embedding layer.



(a) Revealing Game

(b) Hiding Game

Figure 7. SST-2 binary sentiment classification dataset.



(a) Revealing Game

(b) Hiding Game

Figure 8. AG News four topic classification dataset.

Additional results for Table 1.

Table 2. Area Under the Curve for Revealing Game (↑ is better) and Hiding Game (↓ is better)

Explainer	Revealing	Hiding
Simple	0.799	0.756
Decoded Simple l_5	0.816	**0.687**
Decoded Simple l_6	0.828	0.694
Decoded Simple l_7	**0.835**	0.689

(a) SST-2 Dataset (Figure 7)

Explainer	Revealing	Hiding
Simple l_0	0.883	0.765
Decoded Simple l_5	0.866	0.762
Decoded Simple l_6	**0.891**	**0.716**
Decoded Simple l_7	0.847	0.774

(b) AG News Dataset (Figure 8)

A.3. Word Clouds

The size of the tokens in the word clouds are reflective of the weighted tokens scores.

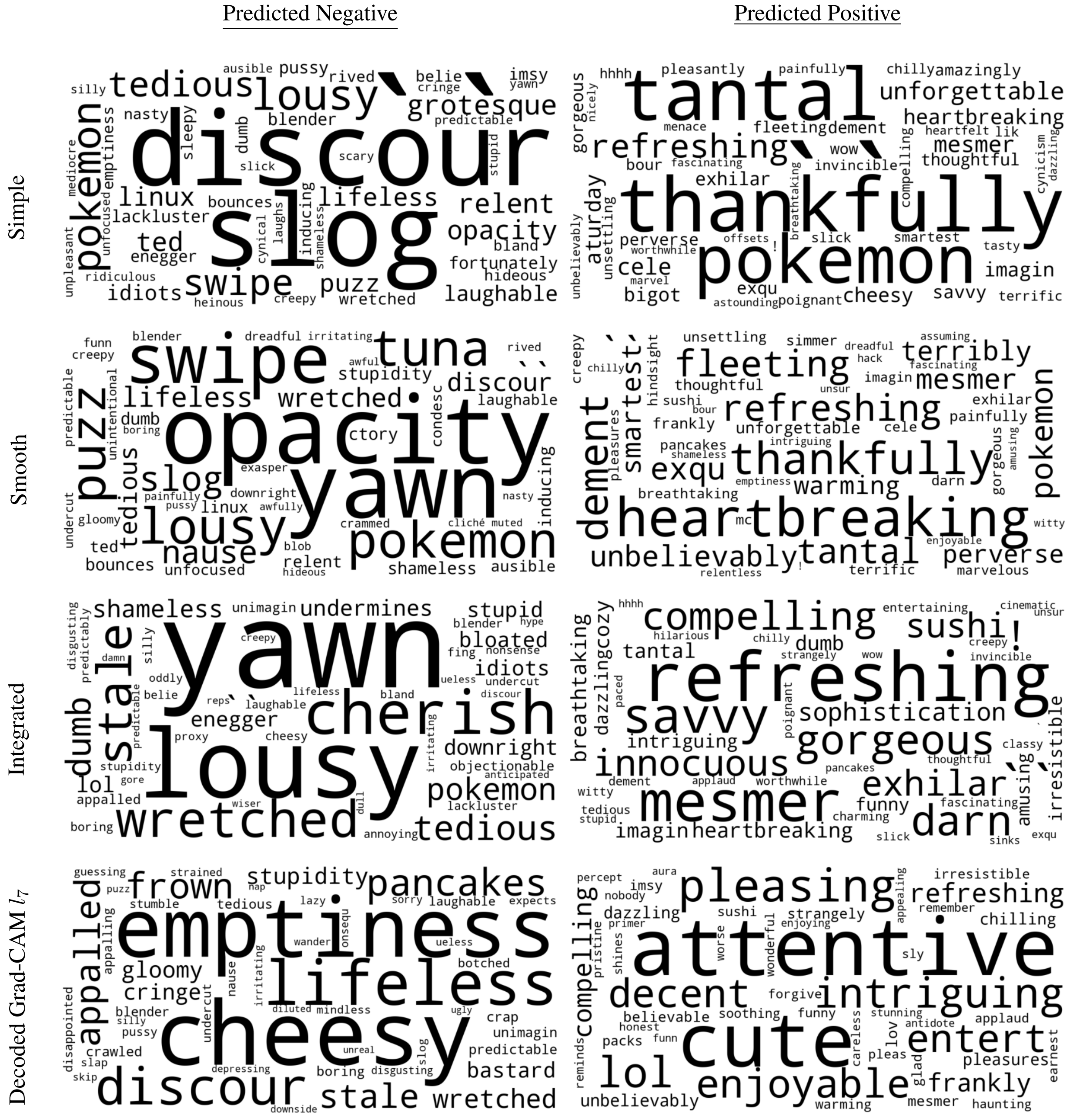


Figure 9. Word Clouds with Top 50 Tokens for SST-2 dataset.

While the left column (Predicted Negative) of Figure 9 is composed of generally negative terms, there are some more puzzling tokens that are deemed important according to the AllenNLP Interpret explainers such as ‘ ‘ and pokemon. The right column (Predicted Positive) also seems to have some tokens with negative connotations such as {menace, dement} in the Simple, {terribly, dement, hack. dreadful} in the Smooth, and {dumb, stupid, tedious} in the Integrated explainer.

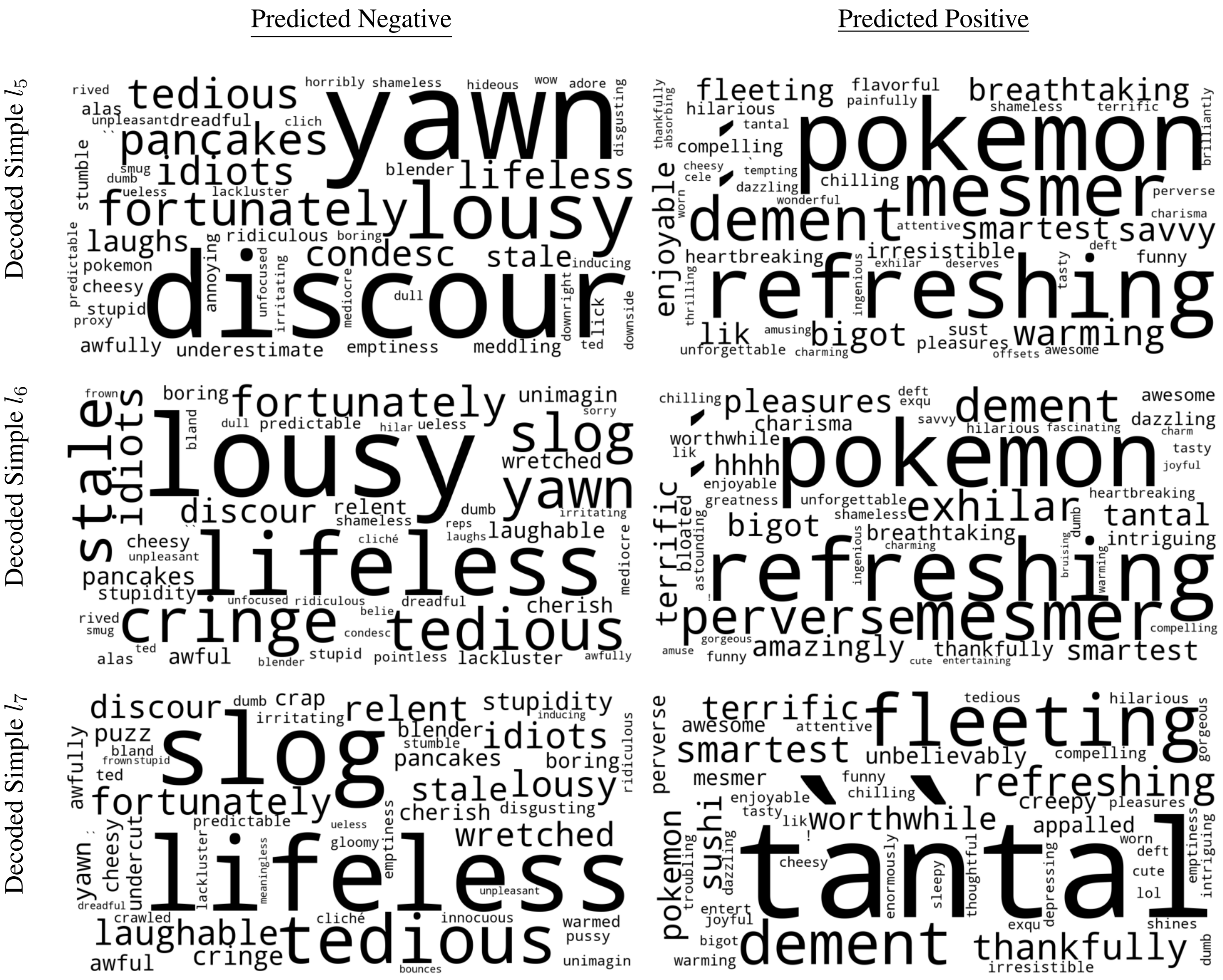


Figure 10. Word Clouds with Top 50 Tokens for SST-2 dataset.

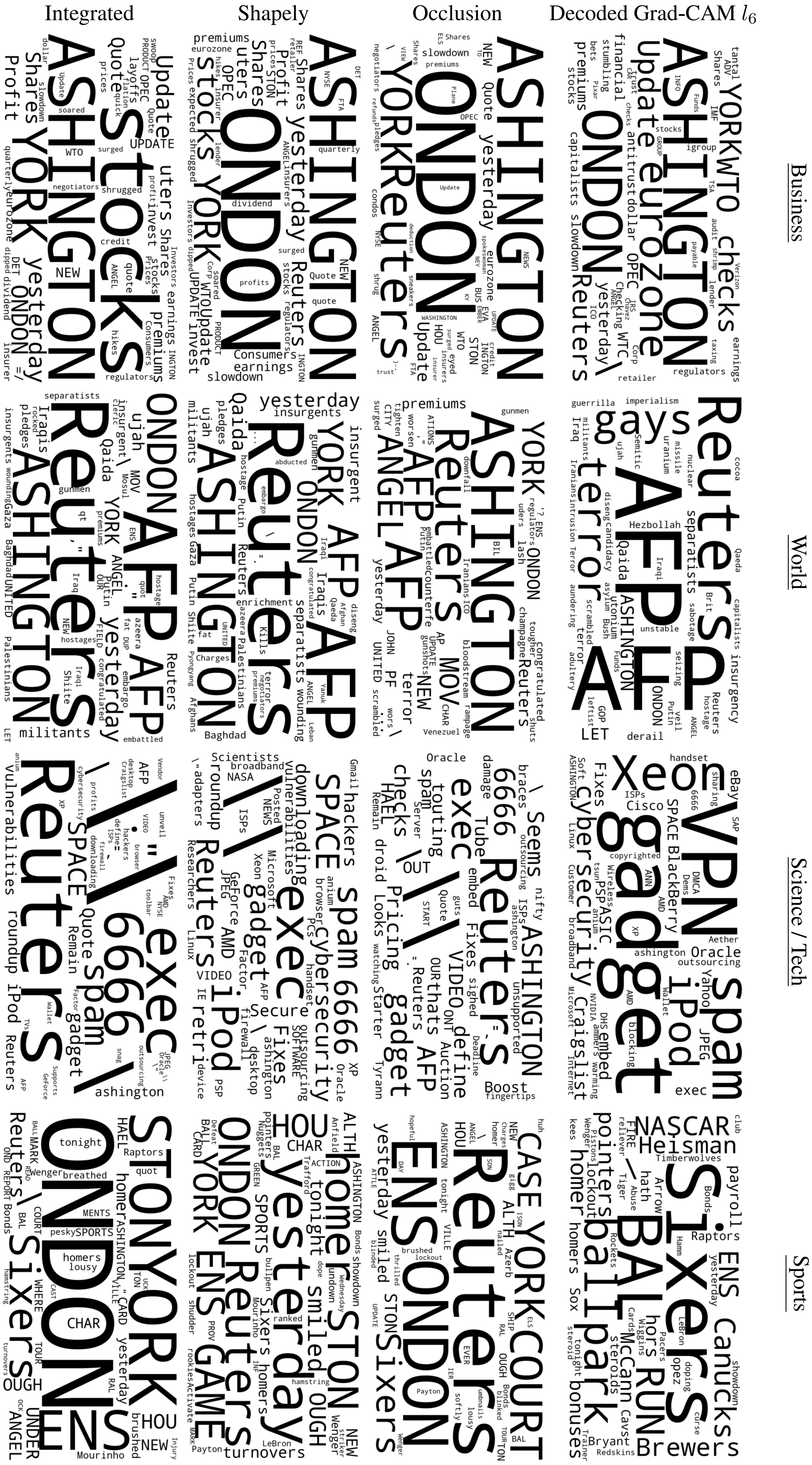


Figure 11: Word Clouds with Top 50 Tokens for AG News dataset

A.4. Overlapping Tokens

Table 3. Tokens in Top 50 appearing in both World and Sports classes of AG News dataset

Explainer	Overlapping Tokens
Shapely	NEW, apologised, FIELD, Defeat, CHAR, ONDON, YORK, UNITED, Thursday, shook, ASHINGTON, Wednesday, Tuesday, ENS, tonight, roared, Reuters, AP, ANGEL, yesterday
Occlusion	NEW, ELS, Calif, AFP, \\, YORK, ONDON, ASHINGTON, Tuesday, ENS, IJ, UPDATE, Charges, awaits, IGH, ,, Monday, roared, Reuters, VER, AP, ANGEL, yesterday, BE
Integrated	NEW, apologised, FIELD, Calif, chilly, \\, ONDON, YORK, Talks, UNITED, Thursday, ASHINGTON, Wednesday, ENS, PARK, Update, embattled, quot, roared, Reuters, AP, ANG, ., ANGEL, yesterday
Decoded Grad-CAM l_6	(none)

Table 4. Tokens in Top 50 appearing in both World and Business classes of AG News dataset

Explainer	Overlapping Tokens
Shapely	NEW, ONDON, YORK, Charges, expected, Thursday, rattled, ASHINGTON, Reuters, Wednesday, premiums, Tuesday, plunged, surged, ANGEL, yesterday
Occlusion	EVA, NEW, ELS, Calif, optimism, \\, ONDON, YORK, ASHINGTON, Tuesday, UPDATE, Charges, Update, Shares, hammered, Monday, surged, embattled, Reuters, MOV, premiums, Profit, regulators, ANGEL, yesterday
Integrated	NEW, negotiators, YORK, ONDON, pledges, Thursday, ASHINGTON, Wednesday, Tuesday, plunged, tighten, Update, IPO, soared, surged, Says, expected, yesterday, premiums, ANGEL, Charges
Decoded Grad-CAM l_6	ONDON, Funds, ASHINGTON, Reuters, capitalists, ANGEL

Table 5. Tokens in Top 50 appearing in both Sports and Business classes of AG News dataset

Explainer	Overlapping Tokens
Shapely	NEW, NEY, STON, ONDON, YORK, Thursday, ASHINGTON, Wednesday, Tuesday, INGTON, eased, OND, woes, HOU, Reuters, ANGEL, MARK, yesterday, rallied
Occlusion	NEW, OCK, BUR, ELS, STON, Calif, \\, ONDON, YORK, roaring, Thursday, ASHINGTON, Wednesday, TON, Tuesday, INGTON, STER, eased, UPDATE, ANC, Charges, –, Monday, HOU, bruised, Reuters, NEWS, BUS, ANGEL, yesterday
Integrated	ANC, NEW, OCK, YORK, ONDON, HOU, Thursday, Update, ASHINGTON, Wednesday, INGTON, STER, TOR, ANGEL, MARK, yesterday, rallied
Decoded Grad-CAM l_6	\\, yesterday

Table 6. Tokens in Top 50 appearing in both World and Sci/Tech classes of AG News dataset

Explainer	**Overlapping Tokens**
Shapely	NEW, ONDON, YORK, Charges, expected, Thursday, rattled, ASHINGTON, Reuters, Wednesday, premiums, Tuesday, plunged, surged, ANGEL, yesterday
Occlusion	expected, Update, ASHINGTON, ,”, Reuters, terror, Reuters, AFP, Calls, \\
Integrated	unveil, AFP, Reuters, ,”, Reuters, AFP, .”, ANGEL, \\
Decoded Grad-CAM l_6	ASHINGTON

Table 7. Tokens in Top 50 appearing in both Sports and Sci/Tech classes of AG News dataset

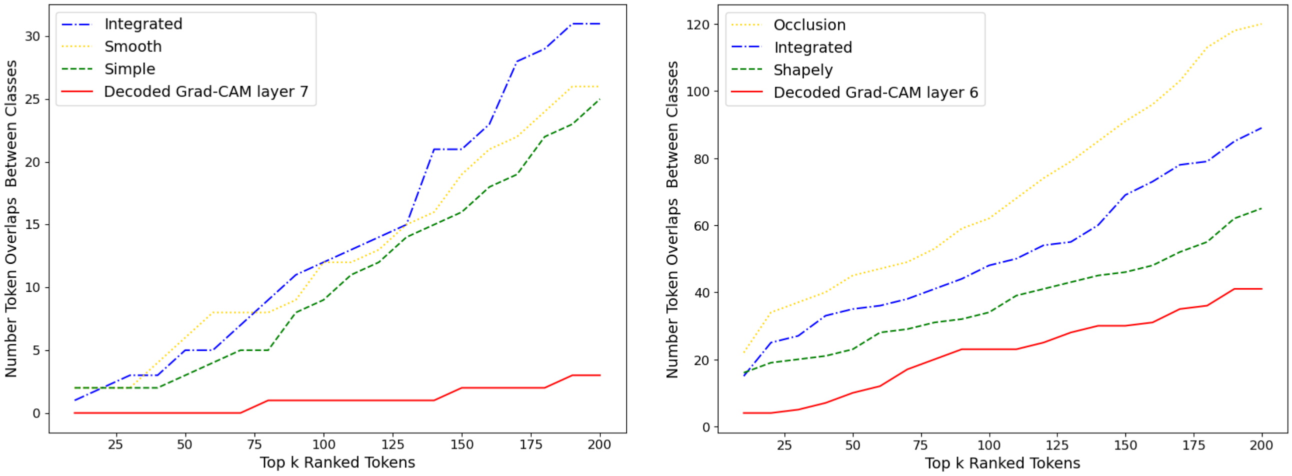
Explainer	**Overlapping Tokens**
Shapely	showdown, Reuters, HAS
Occlusion	ATT, Boost, –, ASHINGTON, ,”, Reuters, HAEL, WITH, STON, AFP, STER, sighed, Factor, \\
Integrated	showdown, Reuters, HAEL, Adds, .”, pesky, ANGEL, \\
Decoded Grad-CAM l_6	(none)

Table 8. Tokens in Top 50 appearing in both Business and Sci/Tech classes of AG News dataset

Explainer	**Overlapping Tokens**
Shapely	Quote, Boost, Reuters, trust
Occlusion	NEY, Update, –, quot, Quote, Consumers, ASHINGTON, HERE, Reuters, STON, STER, BlackBerry, Customers, EMBER, Pact, \\, Update
Integrated	daq, Quote, Craigslist, checks, ?, uters, ANGEL, profits, Update
Decoded Grad-CAM l_6	ASHINGTON

Table 9. Tokens in Top 50 appearing in both classes of SST-2 dataset

Explainer	**Overlapping Tokens**
Simple	slick, ‘ ‘, pokemon
Smooth	pokemon, creepy, shameless, dreadful, painfully, ‘ ‘
Integrated	dumb, creepy, stupid, tedious, ‘ ‘
Decoded Grad-CAM l_7	(none)



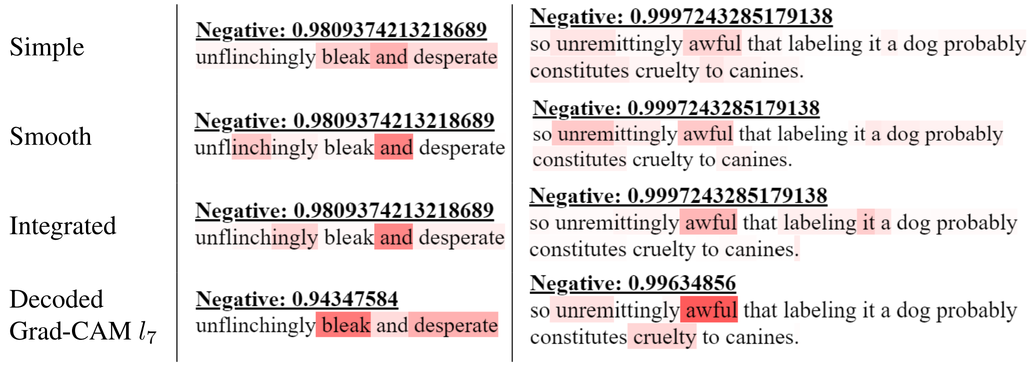
(a) SST-2 dataset

(b) AG News dataset

Figure 12. Number of tokens that appear in multiple classes for the top k most important tokens.

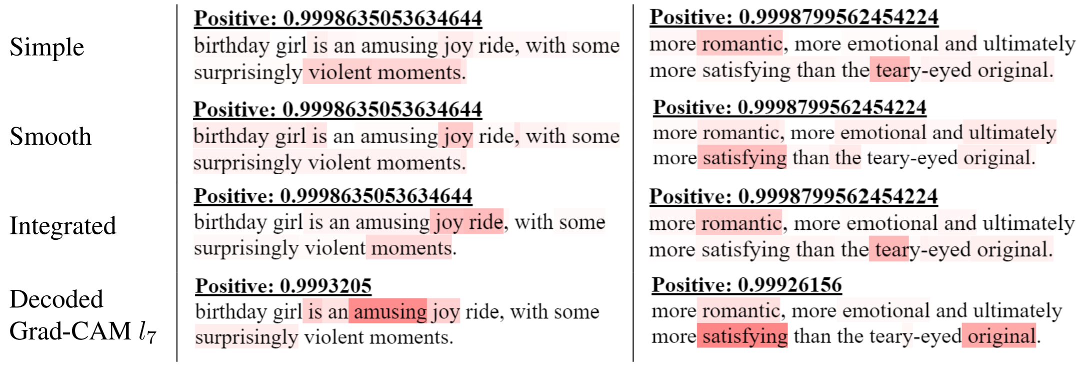
A.5. Examples of Highlighted Explanations

The underlined text above each snippet is the predicted class for that method with corresponding prediction probability and the intensity of the highlighted color reflects the relative importance of each token normalized for each input sequence. All snippets shown are of correctly predicted examples. We have provided highlights of all input sentences in the validation / test sets for both datasets in an attached supplementary file.



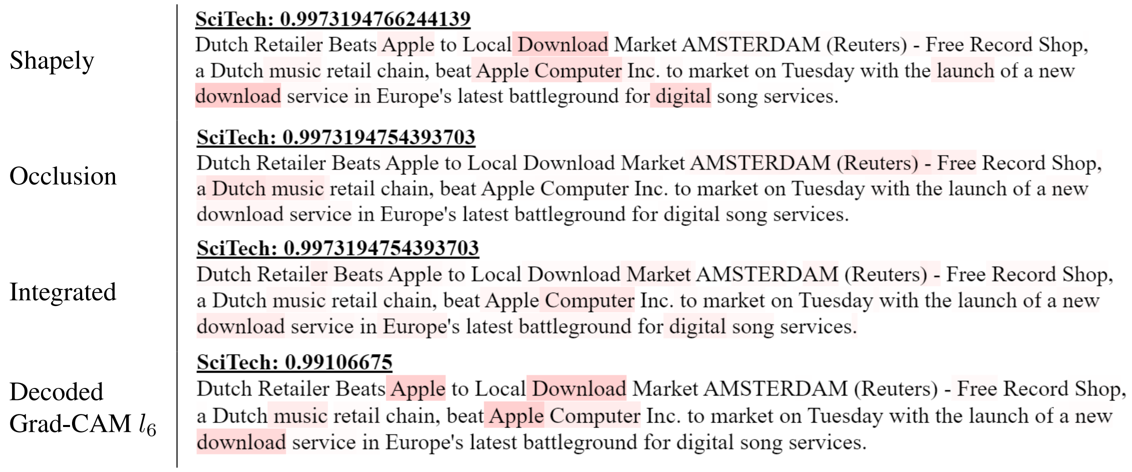
Simple	**Negative: 0.9809374213218689** unflinchingly bleak and desperate	**Negative: 0.9997243285179138** so unremittingly awful that labeling it a dog probably constitutes cruelty to canines.
Smooth	**Negative: 0.9809374213218689** unflinchingly bleak and desperate	**Negative: 0.9997243285179138** so unremittingly awful that labeling it a dog probably constitutes cruelty to canines.
Integrated	**Negative: 0.9809374213218689** unflinchingly bleak and desperate	**Negative: 0.9997243285179138** so unremittingly awful that labeling it a dog probably constitutes cruelty to canines.
Decoded Grad-CAM l_7	**Negative: 0.94347584** unflinchingly bleak and desperate	**Negative: 0.99634856** so unremittingly awful that labeling it a dog probably constitutes cruelty to canines.

Figure 12. STT-2 Example 1: For the left column example, Decoded Grad-CAM l_7 and to an extent AllenNLP Interpret's Simple highlight the negative sentiment words ”bleak” and ”desperate”, but all three of AllenNLP Interpret's methods also focus on ”and”. For the right column example, all four methods focus on ”awful” and ”unrem”(ittingly), but the AllenNLP Interpret's methods are more noisy with highlights on unrelated terms such as ”it”, ”dog” and ”constitutes”.



Simple	**Positive: 0.99986350053634644** birthday girl is an amusing joy ride, with some surprisingly violent moments.	**Positive: 0.9998799562454224** more romantic, more emotional and ultimately more satisfying than the teary-eyed original.
Smooth	**Positive: 0.99986350053634644** birthday girl is an amusing joy ride, with some surprisingly violent moments.	**Positive: 0.9998799562454224** more romantic, more emotional and ultimately more satisfying than the teary-eyed original.
Integrated	**Positive: 0.99986350053634644** birthday girl is an amusing joy ride, with some surprisingly violent moments.	**Positive: 0.9998799562454224** more romantic, more emotional and ultimately more satisfying than the teary-eyed original.
Decoded Grad-CAM l_7	**Positive: 0.9993205** birthday girl is an amusing joy ride, with some surprisingly violent moments.	**Positive: 0.99926156** more romantic, more emotional and ultimately more satisfying than the teary-eyed original.

Figure 12. STT-2 Example 2: For the left column example, Decoded Grad-CAM l_7 focuses strongly on the positive phrases ”is an amusing joy” and ”surprising”. The other methods also highlight these terms, but less clearly with unrelated words such as ”moment” and potentially negative words such as ”violent”. For the right column example, all methods focus on the positive words ”romantic”, ”satisfying”, ”original”, and ”emotional”; however the AllenNLP Interpret's methods are more noisy and highlight many other words too.



Shapely

SciTech: 0.9973194766244139
Dutch Retailer Beats Apple to Local Download Market AMSTERDAM (Reuters) - Free Record Shop, a Dutch music retail chain, beat Apple Computer Inc. to market on Tuesday with the launch of a new download service in Europe's latest battleground for digital song services.

Occlusion

SciTech: 0.9973194754393703
Dutch Retailer Beats Apple to Local Download Market AMSTERDAM (Reuters) - Free Record Shop, a Dutch music retail chain, beat Apple Computer Inc. to market on Tuesday with the launch of a new download service in Europe's latest battleground for digital song services.

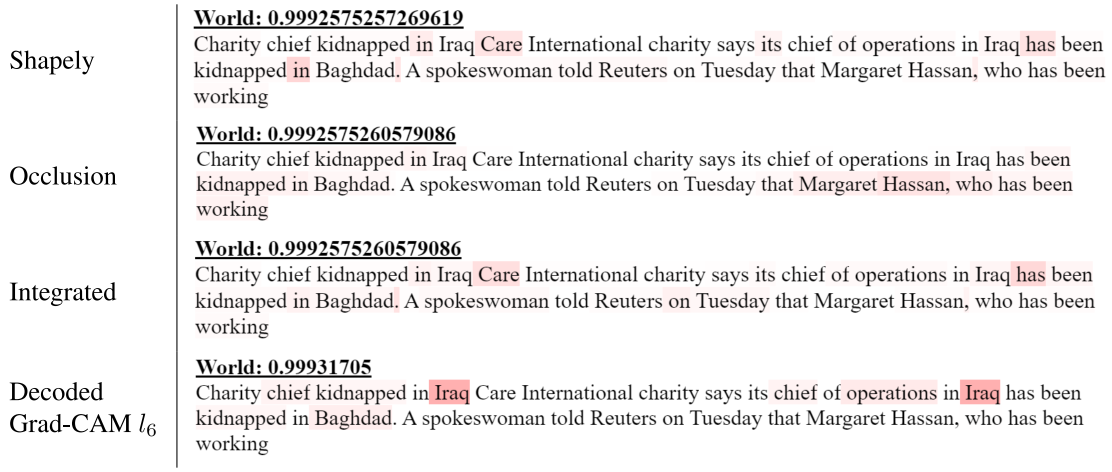
Integrated

SciTech: 0.9973194754393703
Dutch Retailer Beats Apple to Local Download Market AMSTERDAM (Reuters) - Free Record Shop, a Dutch music retail chain, beat Apple Computer Inc. to market on Tuesday with the launch of a new download service in Europe's latest battleground for digital song services.

Decoded Grad-CAM l_6

SciTech: 0.99106675
Dutch Retailer Beats Apple to Local Download Market AMSTERDAM (Reuters) - Free Record Shop, a Dutch music retail chain, beat Apple Computer Inc. to market on Tuesday with the launch of a new download service in Europe's latest battleground for digital song services.

Figure 12. AG News Example 1: Decoded Grad-CAM l_6 and Shapely focus on highlighting ”Apple” (a tech company) along with technology terms like ”Download” and ”Computer”. Occlusion focuses on terms related to the Netherlands such as ”AMSTERDAM” and ”Dutch”, which do not have an obvious connection to technology. Integrated lightly highlights a large number of words, but some are technology related ones.



Shapely

World: 0.9992575257269619
Charity chief kidnapped in Iraq Care International charity says its chief of operations in Iraq has been kidnapped in Baghdad. A spokeswoman told Reuters on Tuesday that Margaret Hassan, who has been working

Occlusion

World: 0.9992575260579086
Charity chief kidnapped in Iraq Care International charity says its chief of operations in Iraq has been kidnapped in Baghdad. A spokeswoman told Reuters on Tuesday that Margaret Hassan, who has been working

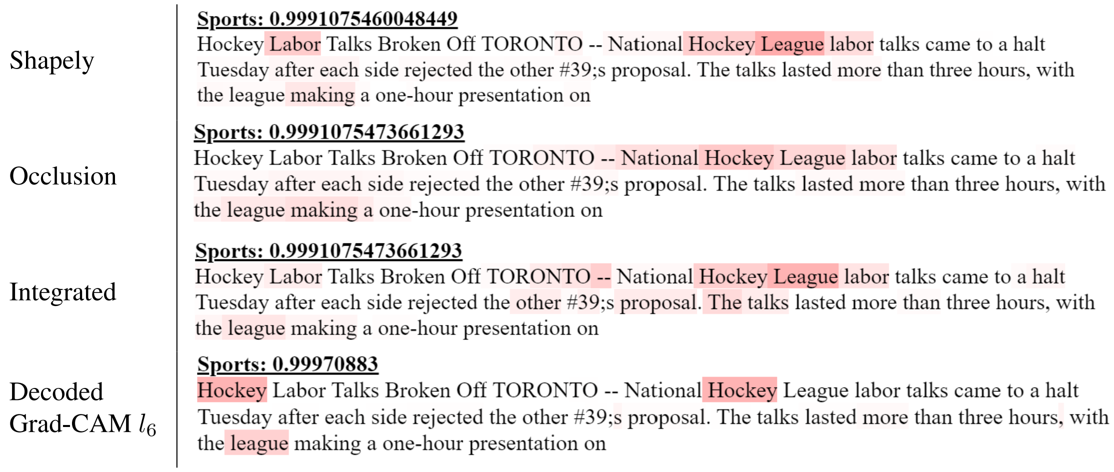
Integrated

World: 0.9992575260579086
Charity chief kidnapped in Iraq Care International charity says its chief of operations in Iraq has been kidnapped in Baghdad. A spokeswoman told Reuters on Tuesday that Margaret Hassan, who has been working

Decoded Grad-CAM l_6

World: 0.99931705
Charity chief kidnapped in Iraq Care International charity says its chief of operations in Iraq has been kidnapped in Baghdad. A spokeswoman told Reuters on Tuesday that Margaret Hassan, who has been working

Figure 12. AG News Example 2: Decoded Grad-CAM l_6 highlights the words ”chief” and ”kidnapped” along with terms related to the Middle East region (”Iraq” and ”Baghdad”). Occlusion lightly highlights the phrases ”kidnapped in Iraq” and ”kidnapped in Baghdad”, which also are meaningful. Shapely and Integrated have less clear explanations with focus on the words ”in”, ”Care”, and the punctuation.



Shapely

Sports: 0.9991075460048449

Hockey Labor Talks Broken Off TORONTO -- National Hockey League labor talks came to a halt Tuesday after each side rejected the other #39;s proposal. The talks lasted more than three hours, with the league making a one-hour presentation on

Occlusion

Sports: 0.9991075473661293

Hockey Labor Talks Broken Off TORONTO -- National Hockey League labor talks came to a halt Tuesday after each side rejected the other #39;s proposal. The talks lasted more than three hours, with the league making a one-hour presentation on

Integrated

Sports: 0.9991075473661293

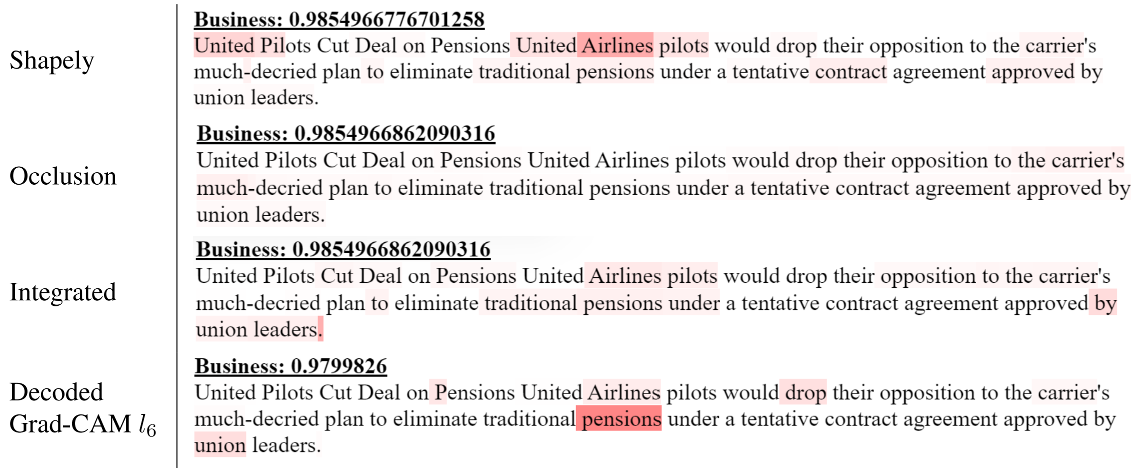
Hockey Labor Talks Broken Off TORONTO -- National Hockey League labor talks came to a halt Tuesday after each side rejected the other #39;s proposal. The talks lasted more than three hours, with the league making a one-hour presentation on

Decoded Grad-CAM l_6

Sports: 0.99970883

Hockey Labor Talks Broken Off TORONTO -- National Hockey League labor talks came to a halt Tuesday after each side rejected the other #39;s proposal. The talks lasted more than three hours, with the league making a one-hour presentation on

Figure 12. AG News Example 3: Decoded Grad-CAM l_6 heavily highlights the word ”Hockey”. The other methods also have some focus on hockey terms such as the phrase ”National Hockey League labor”, but are noisy and also highlight many unrelated terms such as ”The talks” and ”after each”.



Shapely

Business: 0.9854966776701258

United Pilots Cut Deal on Pensions United Airlines pilots would drop their opposition to the carrier's much-decried plan to eliminate traditional pensions under a tentative contract agreement approved by union leaders.

Occlusion

Business: 0.9854966862090316

United Pilots Cut Deal on Pensions United Airlines pilots would drop their opposition to the carrier's much-decried plan to eliminate traditional pensions under a tentative contract agreement approved by union leaders.

Integrated

Business: 0.9854966862090316

United Pilots Cut Deal on Pensions United Airlines pilots would drop their opposition to the carrier's much-decried plan to eliminate traditional pensions under a tentative contract agreement approved by union leaders.

Decoded Grad-CAM l_6

Business: 0.9799826

United Pilots Cut Deal on Pensions United Airlines pilots would drop their opposition to the carrier's much-decried plan to eliminate traditional pensions under a tentative contract agreement approved by union leaders.

Figure 12. AG News Example 4: Decoded Grad-CAM l_6 heavily highlights the word ”pensions” with some additional focus on ”union”; however, it also does highlight some less clear terms such as ”carrier”, ”drop”, and ”Airlines”. Shapely and Integrated also highlight key business terms such as ”traditional pensions”, ”contract”, and ”union leaders”; although Shapely also puts a lot of emphasis on ”United” and Integrated on ”Airlines pilots”. Occlusion lightly highlights everything and does not have an clear explanations.