Local and global model interpretability via backward selection and clustering

Brandon Carter*
MIT CSAIL
bcarter@csail.mit.edu

Jonas Mueller*
MIT CSAIL
jonasmueller@csail.mit.edu

Siddhartha Jain
MIT CSAIL
sj1@csail.mit.edu

David Gifford
MIT CSAIL
gifford@mit.edu

Abstract

Local explanation frameworks aim to rationalize particular decisions made by a black-box prediction model. Existing techniques are often restricted to a specific type of predictor or based on input saliency, which may be undesirably sensitive to factors unrelated to the model's decision making process. We instead propose *sufficient input subsets* that identify minimal subsets of features whose observed values alone suffice for the same decision to be reached, even if all other input feature values are missing. General principles that globally govern a model's decision-making can also be revealed by searching for clusters of such input patterns across many data points. Our approach is conceptually straightforward, entirely model-agnostic, simply implemented using instance-wise backward selection, and able to produce more concise rationales than existing techniques. We demonstrate the utility of our interpretation method on neural network models trained on text and image data.

1 Introduction

The rise of neural networks and nonparametric methods in machine learning (ML) has driven significant improvements in prediction capabilities, while simultaneously earning the field a reputation of producing complex black-box models. Vital applications, which could benefit most from improved prediction, are often deemed too sensitive for opaque learning systems. Consider the widespread use of ML for screening people, including models that deny defendants' bail [1] or reject loan applicants [2]. It is imperative that such decisions can be interpretably rationalized. Interpretability is also crucial in scientific applications, where it is hoped that general principles may be extracted from accurate predictive models [3, 4, 5].

One simple explanation for *why* a particular black-box decision is reached may be obtained via a sparse subset of the input features whose values form the basis for the model's decision – a *rationale*. For text (or image) data, a rationale might consist of a subset of positions in the document (or image) together with the words (or pixel-values) occurring at these positions (see Figures 1 and 7). To ensure interpretations remain fully faithful to an arbitrary model, our rationales do not attempt to summarize the (potentially complex) operations carried out within the model, and instead merely point to the relevant information it uses to arrive at a decision [6]. For high-dimensional inputs, sparsity of the rationale is imperative for greater interpretability.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

*Equal contribution. Code is available at: https://github.com/b-carter/sis_interpretability

Here, we propose a local explanation framework to produce rationales for a learned model that has been trained to map inputs $\mathbf{x} \in \mathcal{X}$ via some arbitrary learned function $f : \mathcal{X} \to \mathbb{R}$. Unlike many other interpretability techniques, our approach is not restricted to vector-valued data and does not require gradients of f. Rather, each input example is solely presumed to have a set of indexable features $\mathbf{x} = [x_1, \ldots, x_p]$, where each $x_i \in \mathbb{R}^d$ for $i \in [p] = \{1, \ldots, p\}$. We allow for features that are unordered (set-valued input) and whose number p may vary from input to input. A rationale corresponds to a sparse subset of these indices $S \subseteq [p]$ together with the specific values of the features in this subset.

To understand why a certain decision was made for a given input example $\mathbf{x}$, we propose a particular rationale called a *sufficient input subset* (SIS). Each SIS consists of a minimal input pattern present in $\mathbf{x}$ that alone suffices for f to produce the same decision, even if provided no other information about the rest of $\mathbf{x}$. Presuming the decision is based on $f(\mathbf{x})$ exceeding some pre-specified threshold $\tau \in \mathbb{R}$, we specifically seek a minimal-cardinality subset S of the input features such that $f(\mathbf{x}_S) \geqslant \tau$. Throughout, we use $\mathbf{x}_S \in \mathcal{X}$ to denote a modified input example in which all information about the values of features outside subset S has been masked with features in S remaining at their original values. Thus, each SIS characterizes a particular standalone input pattern that drives the model toward this decision, providing sufficient justification for this choice from the model's perspective, even without any information on the values of the other features in $\mathbf{x}$.

In classification settings, f might represent the predicted probability of class C where we decide to assign the input to class C if $f(\mathbf{x}) \geqslant \tau$, chosen based on precision/recall considerations. Each SIS in such an application corresponds to a small input pattern that on its own is highly indicative of class C, according to our model. Note that by suitably defining f and τ with respect to the predictor outputs, any particular decision for input $\mathbf{x}$ can be precisely identified with the occurrence of $f(\mathbf{x}) \geqslant \tau$, where higher values of f are associated with greater confidence in this decision.

For a given input $\mathbf{x}$ where $f(\mathbf{x}) \geqslant \tau$, this work presents a simple method to find a complete collection of sufficient input subsets, each satisfying $f(\mathbf{x}_S) \geqslant \tau$, such that there exists no additional SIS outside of this collection. Each SIS may be understood as a disjoint piece of evidence that would lead the model to the same decision, and why this decision was reached for $\mathbf{x}$ can be unequivocally attributed to the SIS-collection. Furthermore, global insight on the general principles underlying the model's decision-making process may be gleaned by clustering the types of SIS extracted across different data points (see Figure 5 and Table 1). Such insights allow us to compare models based not only on their accuracy, but also on human-determined relevance of the concepts they target. Our method's simplicity facilitates its utilization by non-experts who may know very little about the models they wish to interrogate.

2 Related Work

Certain neural network variants such as attention mechanisms [7] and the generator-encoder of [6] have been proposed as powerful yet human-interpretable learners. Other interpretability efforts have tailored decompositions to certain convolutional/recurrent networks [8, 9, 10, 11] , but these approaches are model-specific and only suited for ML experts. Many applications necessitate a model outside of these families, either to ensure supreme accuracy, or if training is done separately with access restricted to a black-box API [12, 13].

An alternative model-agnostic approach to interpretability produces local explanations of f for a particular input $\mathbf{x}$ (e.g. an individual classification decision). Popular local explanation techniques produce attribution scores that quantify the importance of each feature in determining the output of f at $\mathbf{x}$. Examples include LIME, which locally approximates f [14], saliency maps based on f-gradients [15, 16], Layer-wise Relevance Propagation [17], as well as the discrete DeepLIFT approach [5] and its continuous variant – Integrated Gradients (IG), developed to ensure attributions reflect the cumulative difference in f at $\mathbf{x}$ vs. a reference input [18]. A separate class of input-signal-based explanation techniques such as DeConvNet [19], Guided Backprop [20], and PatternNet [21] employ gradients of f in order to identify input patterns that cause f to output large values. However, many such gradient-based saliency methods have been found unreliable, depending not only on the learned function f, but also on its specific architectural implementation and how inputs are scaled [22, 21]. More similar to our approach are recent techniques [23, 24, 25] which also aim to identify input

patterns that best explain certain decisions, but additionally require either a predefined set of such patterns or an auxiliary neural network trained to identify them.

In comparison with the aforementioned methods, our SIS approach presented here is conceptually simple, completely faithful to any type of model, requires no access to gradients of f, requires no additional training of the underlying model f, and does not require training any auxiliary explanation model. Also related to our subset-selection methodology are the ideas of Li et al. [26] and Fong & Veldadi [27], which for a particular input example aim to identify a minimal subset of features whose deletion causes a substantial drop in f such that a different decision would be reached. However, this objective can undesirably produce adversarial artifacts that are not easy to interpret [27]. In contrast, we focus on identifying disjoint minimal subsets of input features whose values suffice to ensure f outputs significantly positive predictions, even in the absence of any other information about the rest of the input. While the techniques used in [26, 27] produce rationales that remain strongly dependent on the rest of the input outside of the selected feature subset, each rationale revealed by our SIS approach is independently considered by f as an entirely sufficient justification for a particular decision in the absence of other information.

3 Methods

Our approach to rationalizing why a particular black-box decision is reached only applies to input examples $\mathbf{x} \in \mathcal{X}$ that meet the decision criterion $f(\mathbf{x}) \geqslant \tau$. For such an input $\mathbf{x}$, we aim to identify a SIS-collection of disjoint feature subsets $S_1, \ldots, S_K \subseteq [p]$ that satisfy the following criteria:

(1) $f(\mathbf{x}_{S_k}) \geqslant \tau$ for each $k = 1, \ldots, K$

(2) There exists no feature subset $S' \subset S_k$ for some $k = 1, \ldots, K$ such that $f(\mathbf{x}_{S'}) \geqslant \tau$

(3) $f(\mathbf{x}_R) < \tau$ for $R = [p] \setminus \bigcup_{k=1}^{K} S_k$ (the remaining features outside of the SIS-collection)

Criterion (1) ensures that for any SIS S_k, the values of the features in this subset alone suffice to justify the decision in the absence of any information regarding the values of the other features. To ensure information that is not vital to reach the decision is not included within the SIS, criterion (2) encourages each SIS to contain a minimal number of features, which facilitates interpretability. Finally, we require that our SIS-collection satisfies a notion of completeness via criterion (3), which states that the same decision is no longer reached for the input after the entire SIS-collection has been masked. This implies the remaining feature values of the input no longer contain sufficient evidence for the same decision. Figures 2 and 6 show SIS-collections found in text/image inputs.

Recall that $\mathbf{x}_S \in \mathcal{X}$ denotes a modified input in which the information about the values of features outside subset S is considered to be missing. We construct $\mathbf{x}_S$ as new input whose values on features in S are identical to those in the original $\mathbf{x}$, and whose remaining features $x_i \in [p] \backslash S$ are each replaced by a special mask $z_i \in \mathbb{R}^{d_i}$ used to represent a missing observation. While certain models are specially adapted to handle inputs with missing observations [28], this is generally not the case. To ensure our approach is applicable to all models, we draw inspiration from data imputation techniques which are a common way to represent missing data [29].

Two popular strategies include hot-deck imputation, in which unobserved values are sampled from their marginal feature distribution, and mean imputation, in which each z_i simply fixed to the average value of feature i in the data. Note that for a linear model, these two strategies are expected to produce an identical change in prediction $f(\mathbf{x}) - f(\mathbf{x}_S)$. We find in practice that the change in predictions resulting from either masking strategy is roughly equivalent even for nonlinear models such as neural networks (Figure S1). In this work, we favor the mean-imputation approach over sampling-based imputation, which would be computationally-expensive and nondeterministic (undesirable for facilitating interpretability). One may also view $\mathbf{z}$ as the *baseline* input value used by feature attribution methods [18, 5], a value which should not lead to particularly noteworthy decisions. Since our interests primarily lie in rationalizing atypical decisions, the average input arising from mean imputation serves as a suitable baseline. Zeros have also been used to mask image/categorical data [26], but empirically, this mask appears undesirably more informative than the mean (predictions more affected by zero-masking).

For an arbitrarily complex function f over inputs with many features p, the combinatorial search to identify sets which satisfy objectives (1)-(3) is computationally infeasible. To find a SIS-collection in

SIScollection(f, $\mathbf{x}$, τ)

1 $S = [p]$
2 **for** $k = 1, 2, \ldots$ **do**
3 $\quad R =$ **BackSelect**$(f, \mathbf{x}, S)$
4 $\quad S_k =$ **FindSIS**$(f, \mathbf{x}, \tau, R)$
5 $\quad S \leftarrow S \backslash S_k$
6 $\quad$ **if** $f(\mathbf{x}_S) < \tau$:
7 $\quad\quad$ **return** S_1,...,S_{k-1}

BackSelect(f, $\mathbf{x}$, S)

1 $R =$ empty stack
2 **while** $S \neq \varnothing$ **do**
3 $\quad i^* = \underset{i \in S}{\operatorname{argmax}}\, f(\mathbf{x}_{S \backslash \{i\}})$
4 $\quad$ Update $S \leftarrow S \backslash \{i^*\}$
5 $\quad$ Push i^* onto top of R
6 **return** R

FindSIS(f, $\mathbf{x}$, τ, R)

1 $S = \varnothing$
2 **while** $f(\mathbf{x}_S) < \tau$ **do**
3 $\quad$ Pop i from top of R
4 $\quad$ Update $S \leftarrow S \cup \{i\}$
5 **if** $f(\mathbf{x}_S) \geqslant \tau$: **return** S
6 **else**: **return** *None*

practice, we employ a straightforward backward selection strategy, which is here applied separately on an example-by-example basis (unlike standard statistical tools which perform backward selection globally to find a fixed set of features for all inputs). The **SIScollection** algorithm details our straightforward procedure to identify disjoint SIS subsets that satisfy (1)-(3) approximately (as detailed in §3.1) for an input $\mathbf{x} \in \mathcal{X}$ where $f(\mathbf{x}) \geqslant \tau$.

Our overall strategy is to find a SIS subset S_k (via **BackSelect** and **FindSIS**), mask it out, and then repeat these two steps restricting each search for the next SIS solely to features disjoint from the currently found SIS-collection $S_1, \ldots, S_k$, until the decision of interest is no longer supported by the remaining feature values. In the **BackSelect** procedure, $S \subset [p]$ denotes the set of remaining unmasked features that are to be considered during backward selection. For the current subset S, step 3 in **BackSelect** identifies which remaining feature $i \in S$ produces the *minimal* reduction in $f(\mathbf{x}_S) - f(\mathbf{x}_{S \backslash \{i\}})$ (meaning it least reduces the output of f if additionally masked), a question trivially answered by running each of the remaining possibilities through the model. This strategy aims to gradually mask out the least important features in order to reveal the core input pattern that is perceived by the model as sufficient evidence for its decision. Finally, we build our SIS up from the last ℓ features omitted during the backward selection, selecting a ℓ value just large enough to meet our sufficiency criterion (1). Because this approach always queries a prediction over the joint set of remaining features S, it is better suited to account for interactions between these features and ensure their sufficiency (i.e. that $f(\mathbf{x}_S) \geqslant \tau$) compared to a forward selection in the opposite direction which builds the SIS upwards one feature at a time by greedily maximizing marginal gains. Throughout its execution, **BackSelect** attempts to maintain the sufficiency of $\mathbf{x}_S$ as the set S shrinks.

3.1 Properties of the SIS-collection

Given p input features, our algorithm requires $\mathcal{O}(p^2 k)$ evaluations of f to identify k SIS, but we can achieve $\mathcal{O}(pk)$ by parallelizing each argmax in **BackSelect** (e.g. batching on GPU). Throughout, let $S_1, \ldots, S_K$ denote the output of **SIScollection** when applied to a given input $\mathbf{x}$ for which $f(\mathbf{x}) \geqslant \tau$. Disjointness of these sets is crucial to ensure computational tractability and that the number of SIS per example does not grow huge and hard to interpret. Proposition 1 below proves that each SIS produced by our procedure will satisfy an approximate notion of minimality. Because we desire minimality of the SIS as specified by (2), it is not appropriate to terminate the backward elimination in **BackSelect** as soon as the sufficiency condition $f(\mathbf{x}_S) \geqslant \tau$ is violated, due to the possible presence of local minima in f along the path of subsets encountered during backward selection (as shown in Figure S24).

Proposition 2 additionally guarantees that masking out the entirety of the feature values in the SIS-collection will ensure the model makes a different decision. Given $f(\mathbf{x}) \geqslant \tau$, it is thus necessarily the case that the observed values responsible for this decision lie within the SIS-collection $S_1, \ldots, S_K$. We point out that for an easily reached decision, where $f(\mathbf{z}) \geqslant \tau$ (i.e. this decision is reached even for the average input), our approach will not output any SIS. Because this same decision would likely be anyway reached for a vast number of inputs in the training data (as a sort of default decision), it is conceptually difficult to grasp what particular aspect of the given $\mathbf{x}$ is responsible.

Proposition 1. *There exists no feature i in any set $S_1, \ldots, S_K$ that can be additionally masked while retaining sufficiency of the resulting subset (i.e. $f(\mathbf{x}_{S_k \backslash \{i\}}) < \tau$ for any $k = 1, \ldots, K, i \in S_k$). Also, among all subsets S considered during the backward selection phase used to produce S_k, this set has the smallest cardinality of those which satisfy $f(\mathbf{x}_S) \geqslant \tau$.*

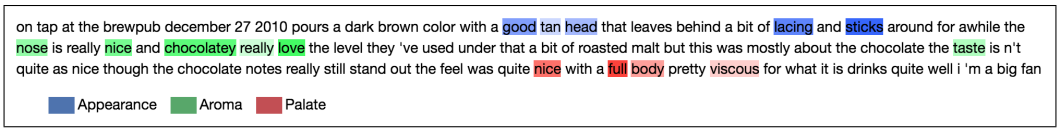


Figure 1: Beer review with one sufficient input subset identified for the prediction of each aspect.

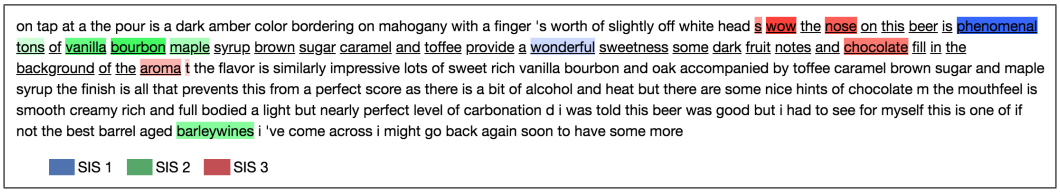


Figure 2: Beer review with three disjoint SIS S_1, S_2, S_3 identified for a positive aroma prediction. Underlined are sentences that human labelers manually annotated as capturing the aroma sentiment.

Proposition 2. *For $\mathbf{x}_{[p]\setminus S*}$, modified by masking all features in the entire SIS-collection $S^* = \bigcup_{k=1}^{K} S_k$, we must have: $f(\mathbf{x}_{[p]\setminus S*}) < \tau$ when $S^* \neq [p]$.*

Unfortunately, nice assumptions like convexity/submodularity are inappropriate for estimated functions in ML. We present various simple forms of practical decision functions for which our algorithms are guaranteed to produce desirable explanations. Example 1 considers interpreting functions of a generalized linear form, Examples 2 & 3 describe functions whose operations resemble generalized logical *OR* & *AND* gates, and Example 4 considers functions that seek out a particular input pattern. Note that features ignored by f are always masked in our backward selection and thus never appear in the resulting SIS-collection.

Example 1. *Suppose the input data are vectors and $f(\mathbf{x}) = g(\beta^T\mathbf{x} + \beta_0)$, where g is monotonically increasing. We also presume $\tau > g(\beta_0)$ and the data were centered such that each feature has mean zero (for ease of notation). In this case, $S_1, \ldots, S_K$ must satisfy criteria (1)-(3). S_1 will consist of the features whose indices correspond to the largest ℓ entries of $\{\beta_1 x_1, \ldots, \beta_p x_p\}$ for some suitable ℓ that depends on the value of τ. It is also guaranteed that $f(\mathbf{x}_{S_1}) \geqslant f(\mathbf{x}_S)$ for any subset $S \subseteq [p]$ of the same cardinality $|S| = \ell$. For each individual feature i where $g(\beta_i x_i + \beta_0) \geqslant \tau$, there will be exist a corresponding SIS S_k consisting only of $\{i\}$. No SIS will include features whose coefficient $\beta_i = 0$, or those whose difference between the observed and average value z_i ($= 0$ here) is of an opposite sign than the corresponding model coefficient (i.e. $\beta_i(x_i - z_i) \leqslant 0$).*

Example 2. *Let $f(\mathbf{x}) = \max\{g_1(\mathbf{x}_{S'_1}), \ldots, g_L(\mathbf{x}_{S'_L})\}$ for some disjoint $S'_1, \ldots, S'_L \subset [p]$ and functions $g_1, \ldots, g_L$, such that for the given $\mathbf{x}$ and threshold τ: $g_1(\mathbf{x}_{S'_1}) > \cdots > g_L(\mathbf{x}_{S'_L}) \geqslant \tau$ and $g_k(\mathbf{x}_{S'_k \setminus \{i\}}) < \tau$ for each $1 \leqslant k \leqslant L, i \in S'_k$. Such f might be functions that model strong interactions between the features in each S_k or look for highly specific value patterns to occur these subsets. In this case,* **SIScollection** *will return L sets such that $S_1 = S'_1, S_2 = S'_2, \ldots, S_L = S'_L$.*

Example 3. *If $f(\mathbf{x}) = \min\{g_1(\mathbf{x}_{S'_1}), \ldots, g_L(\mathbf{x}_{S'_L})\}$ and the same conditions from Example 2 are met, then* **SIScollection** *will return a single set $S_1 = \bigcup_{k=1}^{L} S'_k$.*

Example 4. *Suppose $\mathbf{x} \in \mathbb{R}^p$ with $f(\mathbf{x}) = h(||\mathbf{x}_S - \mathbf{c}_S||)$ where h is monotonically decreasing and $\mathbf{c}_S$ specifies a fixed pattern of input values for features in a certain subset S. For input $\mathbf{x}$ and threshold choice $\tau = f(\mathbf{x})$,* **SIScollection** *will return a single set $S_1 = \{i \in S : |x_i - c_i| < |z_i - c_i|\}$.*

4 Results

We apply our methods to analyze neural networks for text and image data. **SIScollection** is compared with alternative subset-selection methods for producing rationales (see descriptions in Supplement §S1). Note that our **BackSelect** procedure determines an ordering of elements, R, subsequently used to construct the SIS. Depictions of each SIS are shaded based on the feature order in R (darker = later), which can indicate relative feature importance within the SIS.

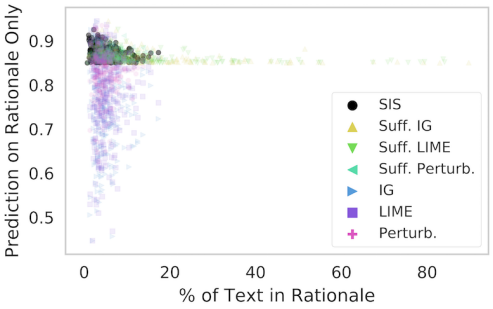


Figure 3: Prediction on rationales only vs. rationale length for various methods in reviews with positive aroma prediction ($\tau = 0.85$).

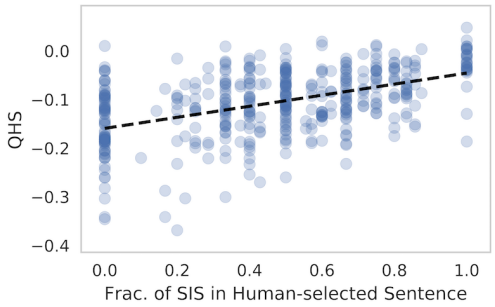


Figure 4: QHS vs. similarity between SIS & annotation in the reviews with positive aroma sentiment (Pearson $\rho = 0.491$, p-value $= 1.5\text{e}{-25}$).

In the "Suff. IG," "Suff. LIME," and "Suff. Perturb." (*sufficiency constrained*) methods, we instead compute the ordering of elements R according to the feature attribution values output by integrated gradients [18], LIME [14], or a perturbative approach that measures the change in prediction when individually masking each feature (see §S1). The rationale subset S produced under each method is subsequently assembled using **FindSIS** exactly as in our approach and thus is guaranteed to satisfy $f(\mathbf{x}_S) \geqslant \tau$. In the "IG," "LIME," and "Perturb." (*length constrained*) methods, we use the same previously described ordering R, but always select the same number of features in the rationale as in the SIS produced by our method (per example).

4.1 Sentiment Analysis of Reviews

We first consider a dataset of beer reviews from BeerAdvocate [30]. Taking the text of a review as input, different LSTM networks [31] are trained to predict user-provided numerical ratings of aspects like aroma, appearance, and palate (details in §S2). Figure 1 shows a sample beer review where we highlight the SIS identified for the LSTM that predicts each aspect. Each SIS only captures sentiment toward the relevant aspect. Figure 2 depicts the SIS-collection identified from a review the LSTM decided to flag for positive aroma.

Figure 3 shows that when the alternative methods described in §4 are length constrained, the rationales they produce often badly fail to meet our sufficiency criterion. Thus, even though the same number of feature values are preserved in the rationale and these alternative methods select the features to which they have assigned the largest attribution values, their rationales lead to significantly reduced f outputs compared to our SIS subsets. If the sufficiency constraint is instead enforced for these alternative methods, the rationales they identify become significantly larger than those produced by **SIScollection**, and also contain many more unimportant features (Table S2, Figure S2).

Benchmarking interpretability methods is difficult because a learned f may behave counterintuitively such that seemingly unreasonable model explanations are in fact faithful descriptions of a model's decision-making process. For some reviews, a human annotator has manually selected which sentences carry the relevant sentiment for the aspect of interest, so we treat these annotations as an alternative rationale for the LSTM prediction. For a review $\mathbf{x}$ whose true and predicted aroma exceed our decision threshold, we define the *quality of human-selected sentences for model explanation* $\text{QHS} = f(\mathbf{x}_S) - f(\mathbf{x})$ where S is the human-selected-subset of words in the review (see examples in Figure S7). High variability of QHS in the annotated reviews (Figure 4) indicates the human rationales often do not contain sufficient information to preserve the LSTM's decision. Figure 4 shows the LSTM makes many decisions based on different subsets of the text than the parts that humans find appropriate for this task. Reassuringly, our SIS more often lie within the selected annotation for reviews with high QHS scores.

4.2 MNIST Digit Classification

We also study a 10-way CNN classifier trained on the MNIST handwritten digits data [32]. Here, we only consider predicted probabilities for one class of interest at a time and always set $\tau = 0.7$ as the

probability threshold for deciding that an image belongs to the class. We extract the SIS-collection from all corresponding test set examples (details in §S3). Example images and corresponding SIS-collections are shown in Figures 6, 7, and S27. Figure 6a illustrates how the SIS-collection drastically changes for an example of a correctly-classified 9 that has been adversarially manipulated [33] to become confidently classified as the digit 4. Furthermore, these SIS-collections immediately enable us to understand why certain misclassifications occur (Figure 6b).

4.3 Clustering SIS for General Insights

Identifying the different input patterns that justify a decision can help us better grasp the general operating principles of a model. To this end, we cluster all of the SIS produced by **SIScollection** applied across a large number of data examples that received the same decision. Clustering is done via DBSCAN, a widely applicable algorithm that merely requires specifying pairwise distances between points [34].

We first apply this procedure to the SIS found across all held-out beer reviews (Test-Fold in Table S1) that received positive aroma predictions from our LSTM network. The distance between two SIS is taken as the Jaccard distance between their bag of words representations. Three clusters depicted in Table 1 (rest in Tables S3, S4) reveal isolated phrases that the LSTM associates with positive aromas in the absence of other context.

We also apply DBSCAN clustering to the SIS found across all MNIST test-examples confidently identified by the CNN as a particular class. Pairwise distances are here defined as the *energy distance* [35] over pixel locations between two SIS subsets (see §S3.3). Figure 5 depicts the SIS clusters identified for digit 4 (others in Figure S28). These reveal distinct feature patterns learned by the CNN to distinguish 4 from other digits, which are clearly present in the vast majority of test set images confidently classified as a 4. For example, cluster C_8 depicts parallel slanted lines, a pattern that never occurs in other digits.

The general insights revealed by our SIS-clustering can also be used to compare the operating-behavior of different models. For the beer reviews, we also train a CNN to compare with our existing LSTM (see §S2.6). For MNIST, we train a multilayer perceptron (MLP) and compare to our existing CNN (see §S3.5). Both networks exhibit similar performance in each task, so it is not immediately clear which model would be preferable in practice. Figure 9 shows the SIS extracted under one model are typically insufficient to receive the same decision from the other model, indicating these models base their positive predictions on different evidence.

Table 2 contains results of jointly clustering the SIS extracted from beer reviews with positive aroma predictions under our LSTM or text-CNN. This CNN tends to learn localized (unigram/bigram) word patterns, while the LSTM identifies more complex multi-word interactions that truly seem more relevant to the target aroma value. Many CNN-SIS are simply phrases with universally-positive sentiment, indicating this model is less capable at distinguishing between positive sentiment toward

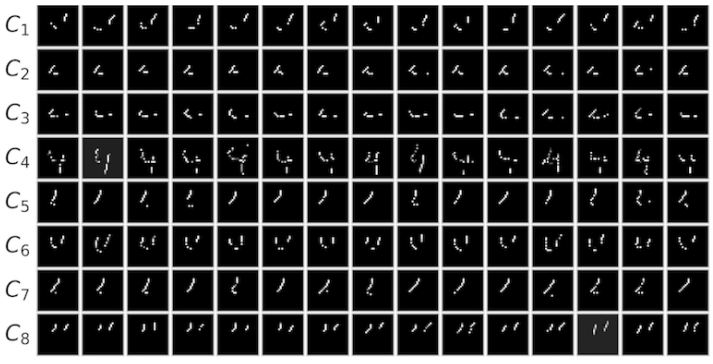
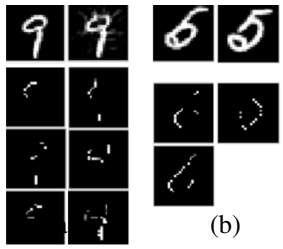


Figure 5: Eight clusters of SIS identified from examples of digit 4. Each row contains fifteen random SIS from a single cluster.



(a) (b)

Figure 6: **(a)** SIS for correctly classified 9 (1st column) and when adversarially perturbed toward class 4 (2nd column). **(b)** SIS for digits 5 that are misclassified as 6 (1st column) and as 0 (2nd column).

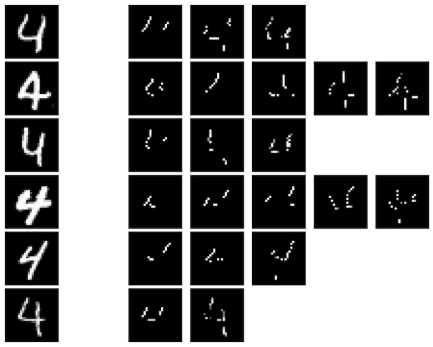


Figure 7: Visualization of SIS-collections for randomly chosen MNIST digits classified as 4 with high confidence by the CNN.

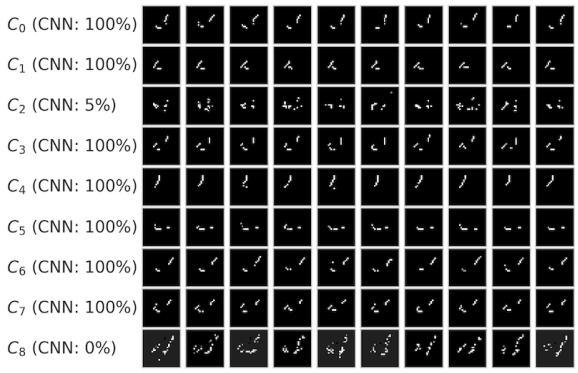


Figure 8: Jointly clustering the MNIST digit 4 SIS from CNN and MLP. We list the percentage of SIS in each cluster stemming from the CNN (rest from MLP).

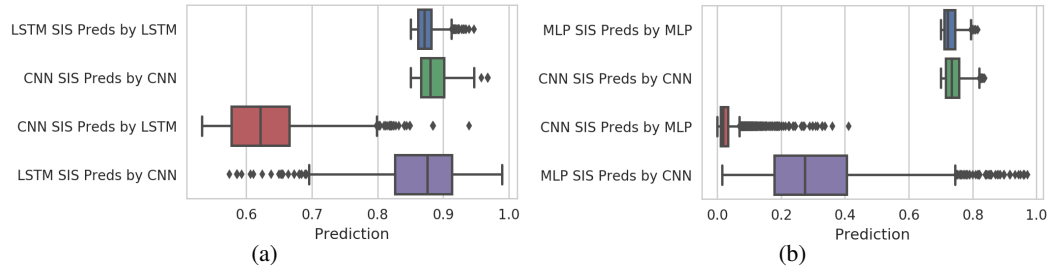


Figure 9: Predictions by one model on the SIS extracted from the other model in: **(a)** beer reviews with positive LSTM/CNN aroma predictions, and **(b)** MNIST digits confidently classified as 4 by CNN/MLP.

aroma vs. other aspects such as taste/look. Figure 8 depicts results from a joint clustering of all SIS extracted from held-out MNIST images confidently classified as a 4 by either the MLP or CNN. Evidently, our MNIST-CNN bases its confidence primarily on spatially-contiguous strokes comprising only a small portion of each digit. MLP-decisions are in contrast based on pixels located throughout the digit, demonstrating this model relies more on the global shape of the handwriting.

Table 1: 3 clusters of SIS extracted from beer reviews with positive CNN aroma predictions. Each row shows 4 most frequent unique SIS in a cluster (each SIS shown as ordered word list with text-positions omitted). Each unique SIS can be present many times in one cluster.

Clu.	SIS #1	SIS #2	SIS #3	SIS #4
C_1	smell amazing wonderful	nice wonderful nose	wonderful amazing	amazing amazing
C_2	grapefruit mango pineapple	pineapple grapefruit pineapple grapefruit	hops grapefruit pineapple floyds	mango pineapple incredible
C_3	creme brulee brulee	creme brulee decadent	incredible creme brulee	creme brulee exceptional

Table 2: Joint clustering of the SIS from beer reviews predicted to have positive aroma by LSTM or CNN. Dashes are used in clusters with under 4 unique SIS. Percentages quantify SIS per cluster from the LSTM.

Clu.	LSTM	SIS #1	SIS #2	SIS #3	SIS #4
C_1	0%	delicious	-	-	-
C_2	0%	very nice	-	-	-
C_3	20%	rich chocolate	very rich	chocolate complex	smells rich
C_4	33%	oak chocolate	chocolate raisins raisins oak bourbon	chocolate oak	raisins chocolate
C_5	70%	complex aroma	aroma complex peaches complex	aroma complex interesting cherries	aroma complex

5 Discussion

This work introduced the idea of interpreting black-box decisions on the basis of sufficient input subsets – minimal input patterns that alone provide sufficient evidence to justify a particular decision. Our methodology is easy to understand for non-experts, applicable to all ML models without any additional training steps, and remains fully faithful to the underlying model without making approximations. While we focus on local explanations of a single decision, clustering the SIS-patterns extracted from many data points reveals insights about a model's general decision-making process. Given multiple models of comparable accuracy, SIS-clustering can uncover critical operating differences, such as which model is more susceptible to spurious training data correlations or will generalize worse to counterfactual inputs that lie outside the data distribution.

Acknowledgments

This work was supported by NIH Grants R01CA218094, R01HG008363, and R01HG008754.

References

[1] Kleinberg J, Lakkaraju H, Leskovec J, Ludwig J, Mullainathan S (2018) Human decisions and machine predictions. *The Quarterly Journal of Economics* 133: 237-293.

[2] Sirignano JA, Sadhwani A, Giesecke K (2018) Deep learning for mortgage risk. *arXiv:160702470* .

[3] Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning. *arXiv:170208608* .

[4] Lipton ZC (2016) The mythos of model interpretability. In: *ICML Workshop on Human Interpretability of Machine Learning*.

[5] Shrikumar A, Greenside P, Kundaje A (2017) Learning important features through propagating activation differences. In: *International Conference on Machine Learning*.

[6] Lei T, Barzilay R, Jaakkola T (2016) Rationalizing neural predictions. In: *Empirical Methods in Natural Language Processing*.

[7] Sha Y, Wang MD (2017) Interpretable predictions of clinical outcomes with an attention-based recurrent neural network. In: *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics*.

[8] Murdoch WJ, Liu PJ, Yu B (2018) Beyond word importance: Contextual decomposition to extract interactions from LSTMs. In: *International Conference on Learning Representations*.

[9] Strobelt H, Gehrmann S, Pfister H, Rush A (2018) LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics* : 667–676.

[10] Olah C, Mordvintsev A, Schubert L (2017) Feature visualization. *Distill* .

[11] Olah C, Satyanarayan A, Johnson I, Carter S, Schubert L, et al. (2018) The building blocks of interpretability. *Distill* .

[12] Caruana R, Lou Y, Gehrke J, Koch P, Sturm M, et al. (2015) Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[13] Tramer F, Zhang F, Juels A, Reiter MK, Ristenpart T (2016) Stealing machine learning models via prediction APIs. In: *USENIX Security Symposium*.

[14] Ribeiro MT, Singh S, Guestrin C (2016) "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.

[15] Baehrens D, Schroeter T, Harmeling S, Kawanabe M, Hansen K, et al. (2010) How to explain individual classification decisions. *Journal of Machine Learning Research* 11: 1803–1831.

[16] Simonyan K, Vedaldi A, Zisserman A (2014) Deep inside convolutional networks: Visualising image classification models and saliency maps. In: *International Conference on Learning Representations*.

[17] Bach S, Binder A, Montavon G, Klauschen F, Müller KR, et al. (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One* 10: e0130140.

[18] Sundararajan M, Taly A, Yan Q (2017) Axiomatic attribution for deep networks. In: *International Conference on Machine Learning*.

[19] Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: *European Conference on Computer Vision*.

[20] Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M (2015) Striving for simplicity: The all convolutional net. In: *International Conference on Learning Representations*.

[21] Kindermans PJ, Schütt KT, Alber M, Müller KR, Erhan D, et al. (2018) Learning how to explain neural networks: PatternNet and PatternAttribution. In: *International Conference on Learning Representations*.

[22] Kindermans PJ, Hooker S, Adebayo J, Alber M, Schütt KT, et al. (2017) The (un) reliability of saliency methods. In: *NIPS Workshop: Interpreting, Explaining and Visualizing Deep Learning - Now what?*

[23] Kim B, Wattenberg M, Gilmer J, Cai C, Wexler J, et al. (2018) Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In: *International Conference on Machine Learning*.

[24] Dabkowski P, Gal Y (2017) Real time image saliency for black box classifiers. In: *Advances in Neural Information Processing Systems*.

[25] Chen J, Song L, Wainwright MJ, Jordan MI (2018) Learning to explain: An information-theoretic perspective on model interpretation. In: *International Conference on Machine Learning*.

[26] Li J, Monroe W, Jurafsky D (2017) Understanding neural networks through representation erasure. *arXiv:161208220* .

[27] Fong RC, Vedaldi A (2017) Interpretable explanations of black boxes by meaningful perturbation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[28] Smola AJ, Vishwanathan S, Hofmann T (2005) Kernel methods for missing variables. In: *Artificial Intelligence and Statistics*.

[29] Rubin DB (1976) Inference and missing data. *Biometrika* 63: 581–592.

[30] McAuley J, Leskovec J, Jurafsky D (2012) Learning attitudes and attributes from multi-aspect reviews. In: *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, pp. 1020–1025.

[31] Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural computation* 9: 1735–1780.

[32] LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86: 2278–2324.

[33] Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. In: *IEEE Symposium on Security and Privacy*.

[34] Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*.

[35] Rizzo ML, Székely GJ (2016) Energy distance. *Wiley Interdisciplinary Reviews: Computational Statistics* 8: 27–38.

Supplementary Information for: Local and global model interpretability via backward selection and clustering

Contents

S1 Detailed Description of Alternative Methods **3**

S2 Details of the Beer Reviews Sentiment Analysis **3**

S2.1 Beer Reviews Data Description . . . 3
S2.2 Model Architecture and Training . . . 3
S2.3 Imputation Strategies: Mean vs. Hot-deck . . . 4
S2.4 Feature Importance Scores . . . 4
S2.5 Additional Results for Aroma aspect . . . 5
S2.6 Understanding Differences Between Sentiment Predictors . . . 6
S2.7 Results for Appearance and Palate aspects . . . 10

S3 Details of the MNIST Analysis **15**

S3.1 Dataset and Model . . . 15
S3.2 Local Minima in Backward Selection . . . 15
S3.3 Energy Distance Between Image SIS . . . 15
S3.4 SIS Clustering and Adversarial Analysis . . . 16
S3.5 Understanding Differences Between MNIST Classifiers . . . 19

List of Figures

1 Beer review with SIS for each aspect . . . 5
2 Beer review with 3 SIS subsets . . . 5
3 Prediction on rationales only vs. rationale length (aroma prediction) . . . 6
4 QHS vs. SIS-annotation similarity . . . 6
5 SIS clusters for digit 4 . . . 7
6 SIS-collections for adversarial and misclassified MNIST digits . . . 7
8 Joint clustering MNIST digit 4 SIS from CNN & MLP . . . 8
9 Predictions on the SIS from alternative models on beer reviews and MNIST digits . . . 8
S1 Mean vs. hot-deck imputation for aroma prediction . . . 5
S2 Feature importance comparison for aroma prediction . . . 5
S3 Length of rationales for aroma prediction . . . 5
S4 Prediction of aroma sentiment on the annotation set . . . 6
S5 Number of SIS per for aroma beer reviews . . . 6
S6 Prediction as function of remaining text for aroma prediction . . . 6
S7 Alignment of human rationales for beer reviews with predictive model . . . 7
S8 Mean vs. hot-deck imputation in appearance prediction . . . 10
S9 Prediction of appearance sentiment on the annotation set . . . 10
S10 Number of SIS per for appearance beer reviews . . . 10
S11 Length of rationales for appearance prediction . . . 10
S12 Feature importance for appearance prediction . . . 10
S13 QHS vs. fraction of SIS in human rationale for appearance prediction . . . 11
S14 Prediction on only rationale vs. rationale length . . . 11

S15 Prediction as function of remaining text for appearance aspect . . . 11
S16 Mean vs. hot-deck imputation for palate prediction . . . 12
S17 Prediction of palate sentiment on the annotation set . . . 12
S18 Number of SIS per for palate beer reviews . . . 12
S19 Length of rationales for palate prediction . . . 12
S20 Feature importance for palate prediction . . . 12
S21 QHS vs. fraction of SIS in human rationale for palate prediction . . . 13
S22 Prediction on only rationale vs. rationale length for palate prediction . . . 13
S23 Prediction as function of remaining text for palate aspect . . . 13
S24 Local Minimum in Backward Selection . . . 15
S25 Number of examples per MNIST digit . . . 16
S26 Number of SIS per MNIST digit . . . 16
S27 SIS examples on MNIST (CNN) . . . 17
S28 SIS clusters identified on MNIST (CNN) . . . 18
S29 SIS clusters identified on MNIST (MLP) . . . 19
S30 SIS examples on MNIST (MLP) . . . 20

List of Tables

1 Three SIS clusters from beer reviews . . . 8
2 Joint clustering of SIS from LSTM and CNN on beer reviews, aroma aspect . . . 8
S1 Summary and performance statistics for beer reviews data . . . 4
S2 Statistics for rationale length and feature importance in aroma prediction . . . 5
S3 SIS clusters for positive aroma prediction . . . 7
S4 SIS clusters for negatives aroma prediction . . . 7
S5 Joint clustering of SIS from LSTM and CNN on beer reviews, positive aroma aspect . . . 9
S6 Joint clustering of SIS from LSTM and CNN on beer reviews, negative aroma aspect . . . 9
S7 Statistics for rationale length and feature importance in appearance prediction . . . 11
S8 SIS clusters for positive appearance prediction . . . 11
S9 SIS clusters for negative appearance prediction . . . 12
S10 Statistics for rationale length and feature importance in palate prediction . . . 13
S11 SIS clusters for positive palate prediction . . . 14
S12 SIS clusters for negative palate prediction . . . 14

S1 Detailed Description of Alternative Methods

In Section 3, we describe a number of alternative methods for identifying rationales for comparison with our method. We use methods based on integrated gradients [36], LIME [37], and feature perturbation. Note that integrated gradients is an attribution method which assigns a numerical score to each input feature. LIME likewise assigns a weight to each feature using a local linear regression model for f around $\mathbf{x}$. In the perturbative approach, we compute the change in prediction when each feature is individually masked, as in Equation 1 (of Section S2.4). Each of these feature orderings R is used to construct a rationale using the **FindSIS** procedure (Section 3) for the "Suff. IG," "Suff. LIME," and "Suff. Perturb." (*sufficiency constrained*) methods.

Note that our text classification architecture (described in Section S2.2) encodes discrete words as 100-dimensional continuous word embeddings. The integrated gradients method returns attribution scores for each coordinate of each word embedding. For each word embedding $x_i \in \mathbf{x}$ (where each $x_i \in \mathbb{R}^{100}$), we summarize the attributions along the corresponding embedding into a single score y_i using the L_1 norm: $y_i = \sum_d |x_{id}|$ and compute the ordering R by sorting the y_i values.

We use an implementation of integrated gradients for Keras-based models from `https://github.com/hiranumn/IntegratedGradients`. In the case of the beer review dataset (Section 4.1), we use the mean embedding vector as a baseline for computing integrated gradients. As suggested in [36], we verified that the prediction at the baseline and the integrated gradients sum to approximately the prediction of the input.

For LIME and our beer reviews dataset, we use the approach described in [37] for textual data, where individual words are removed entirely from the input sequence. We use the implementation of LIME at: `https://github.com/marcotcr/lime`. The `LimeTextExplainer` module is used with default parameters, except we set the maximal number of features used in the regression to be the full input length so we can order all input features.

Additionally, we explore methods in which we use the same ordering R by these alternative methods but select the same number of input features in the rationale to be the median SIS length in the SIS-collection computed by our method on each example: the "IG," "LIME," and "Perturb." (*length constrained*) methods. We compute the feature ordering based on the absolute value of the non-zero integrated gradient attributions. Note that for the length constrained methods, there is no guarantee of sufficiency $f(\mathbf{x}_S) \geqslant \tau$ for any input subset S.

S2 Details of the Beer Reviews Sentiment Analysis

S2.1 Beer Reviews Data Description

As done in [38], we use a preprocessed version of the BeerAdvocate[2] dataset[3] which contains decorrelated numerical ratings toward three aspects: *aroma*, *appearance*, and *palate* (each normalized to $[0, 1]$). Dataset statistics can be found in Table S1. Reviews were tokenized by converting to lowercase and filtering punctuation, and we used a vocabulary containing the top 10,000 most common words. The data also contain subset of human-annotated reviews, in which humans manually selected full sentences in each review that describe the relevant aspects [39]. This annotated set was never seen during training and used solely as part of our evaluation.

S2.2 Model Architecture and Training

Long short-term memory (LSTM) networks are commonly employed for natural language tasks such as sentiment analysis [40, 41]. We use a recurrent neural network (RNN) architecture with two stacked LSTMs as follows:

[2] `https://www.beeradvocate.com/`

[3] `http://snap.stanford.edu/data/web-BeerAdvocate.html`

1. **Input/Embeddings Layer**: Sequence with 500 timesteps, the word at each timestep is represented by a (learned) 100-dimensional embedding
2. **LSTM Layer 1**: 200-unit recurrent layer with LSTM (forward direction only)
3. **LSTM Layer 2**: 200-unit recurrent layer with LSTM (forward direction only)
4. **Dense**: 1 neuron (sentiment output), sigmoid activation

With this architecture, we use the Adam optimizer [42] to minimize mean squared error (MSE) on the training set. We use a held-out set of 3,000 examples for validation (sampled at random from the pre-defined test set used in [38]). Our test set consists of the remaining 7,000 test examples. Training results are shown in Table S1.

Table S1: Summary and performance statistics (mean squared error (MSE) and Pearson correlation coefficient ρ) for beer reviews data and LSTM models.

Aspect	Fold	Size	MSE	Pearson ρ
Appearance	Train	80,000	0.016	0.864
	Validation	3,000	0.024	0.783
	Test	7,000	0.023	0.801
	Annotation	994	0.020	0.563
Aroma	Train	70,000	0.014	0.873
	Validation	3,000	0.024	0.767
	Test	7,000	0.025	0.756
	Annotation	994	0.021	0.598
Palate	Train	70,000	0.016	0.835
	Validation	3,000	0.029	0.680
	Test	7,000	0.028	0.694
	Annotation	994	0.016	0.592

S2.3 Imputation Strategies: Mean vs. Hot-deck

In Section 3, we discuss the problem of masking input features. Here, we show that the mean-imputation approach (in which missing inputs are masked with a mean embedding, taken over the entire vocabulary) produces a nearly identical change in prediction to a nondeterministic hot-deck approach (in which missing inputs are replaced by randomly sampling feature-values from the data). Figure S1 shows the change in prediction $f(\mathbf{x}\backslash\{i\}) - f(\mathbf{x})$ by both imputation techniques after drawing a training example $\mathbf{x}$ and word $x_i \in \mathbf{x}$ (both uniformly at random) and replacing x_i with either the mean embedding or a randomly selected word (drawn from the vocabulary, based on counts in the training corpus). This procedure is repeated 10,000 times. Both resulting distributions have mean near zero ($\mu_{\text{mean-embedding}} = -7.0\text{e}{-4}$, $\mu_{\text{hot-deck}} = -7.4\text{e}{-4}$), and the distribution for mean embedding is slightly narrower ($\sigma_{\text{mean-embedding}} = 0.013$, $\sigma_{\text{hot-deck}} = 0.018$). We conclude that mean-imputation is a suitable method for masking information about particular feature values in our SIS analysis.

We also explored other options for masking word information, e.g. replacement with a zero embedding, replacement with the learned <PAD> embedding, and simply removing the word entirely from the input sequence, but each of these alternative options led to undesirably larger changes in predicted values as a result of masking, indicating they appear more informative to f than replacement via the feature-mean.

S2.4 Feature Importance Scores

For each feature i in the input sequence, we quantify its marginal importance by individually perturbing only this feature:

$$\text{Feature Importance}(i) = \text{prediction on original input} - \text{prediction with feature } i \text{ masked} \quad (1)$$

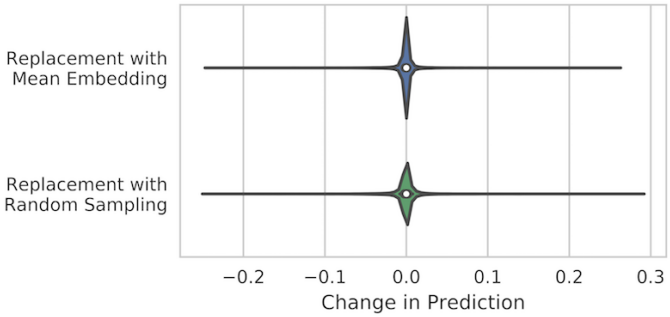


Figure S1: Change in prediction ($f(\mathbf{x}\backslash\{i\}) - f(\mathbf{x})$) after masking a randomly chosen word with mean imputation or hot-deck imputation. 10,000 replacements were sampled from the aroma beer reviews training set.

Table S2: Statistics for rationale length and feature importance in aroma prediction. For rationale length, median and max indicate percentage of input text in the rationale. For marginal perturbed feature importance, we indicate the median importance of features in rationales and features from the other (non-rationale) text. p-values are computed using a Wilcoxon rank-sum test.

Method	Rationale Length (% of text)			Marginal Perturbed Feature Importance		
	Med.	Max	p (vs. SIS)	Med. (Rationale)	Med. (Other)	p (vs. SIS)
SIS	**3.9%**	**17.3%**	–	0.0112	1.50e-05	–
Suff. IG	7.7%	89.7%	5e-26	0.0068	1.85e-05	3e-42
Suff. LIME	7.2%	84.0%	4e-23	0.0075	1.87e-05	1e-35
Suff. Perturb.	5.1%	18.3%	1e-06	0.0209	1.90e-05	1e-72

Note that these marginal Feature Importance scores are identical to those of the Perturb. method described in Section S1. The marginal Feature Importance scores are summarized in Table S2 and Figure S2. Compared to the Suff. IG and Suff. LIME methods, our **SIScollection** technique produces rationales that are much shorter and contain fewer irrelevant (i.e. not marginally important) features (Table S2, Figures S2 and S3). Note that by construction, the rationales of the Suff. Perturb. method contain features with the greatest Feature Importance, since this precisely how the ranking in Suff. Perturb. is defined.

S2.5 Additional Results for Aroma aspect

We apply our method to the set of reviews containing sentence-level annotations. Note that these reviews (and the human annotations) were not seen during training. We choose thresholds $\tau_+ = 0.85$,

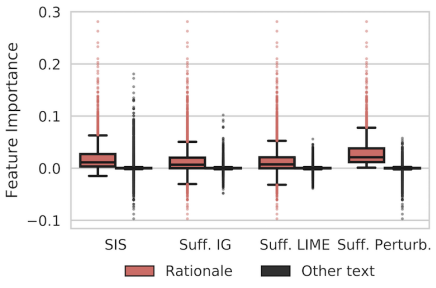


Figure S2: Importance of individual features in the rationales for aroma prediction in beer reviews

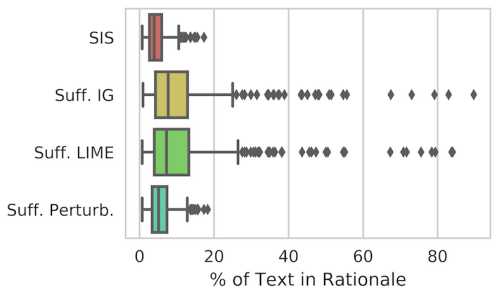


Figure S3: Length of rationales for aroma prediction

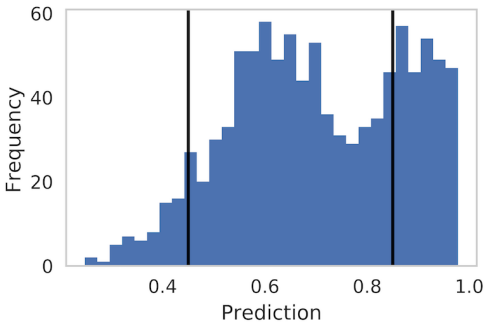


Figure S4: Predictive distribution on the annotation set (held-out) using the LSTM model for aroma. Vertical lines indicate decision thresholds ($\tau_+ = 0.85$, $\tau_- = 0.45$) selected for **SIScollection**.

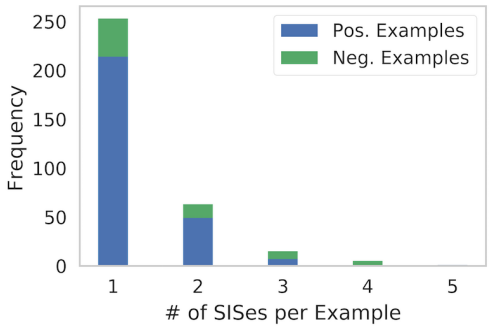


Figure S5: Number of sufficient input subsets for aroma identified by **SIScollection** per example.

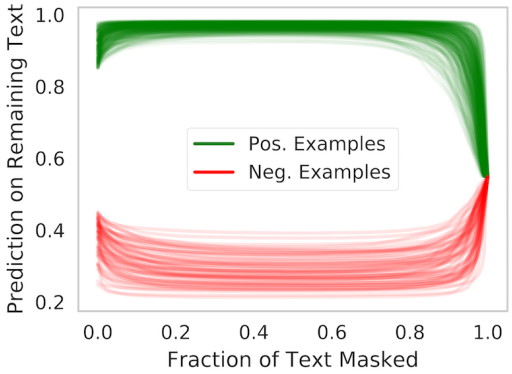


Figure S6: Prediction history on remaining (unmasked) text at each step of the **BackSelect** procedure, for examples where aroma sentiment is predicted.

$\tau_- = 0.45$ for strong positive and strong negative sentiment, respectively, and extract the complete set of sufficient input subsets using our method. Note that in our formulation above, we apply our method to inputs $\mathbf{x}$ where $f(\mathbf{x}) \geqslant \tau$. For the sentiment analysis task, we analogously apply our method for both $f(\mathbf{x}) \geqslant \tau_+$ and $-f(\mathbf{x}) \geqslant -\tau_-$, where the model predicts either strong positive or strong negative sentiment, respectively. These thresholds were set empirically such that they were sufficiently apart, based on the distribution of predictions (Figure S4). For most reviews, **SIScollection** outputs just one or two SIS sets (Figure S5).

We analyzed the predictor output following the elimination of each feature in the **BackSelect** procedure (Section 3). Figure S6 shows the LSTM output on the remaining unmasked text $f(\mathbf{x}_{S\setminus\{i*\}})$ at each iteration of **BackSelect**, for all examples. This figure reveals that only a small number of features are needed by the model in order to make a strong prediction (most features can be removed without changing the prediction). We see that as those final, critical features are removed, there is a rapid, monotonic decrease in output values. Finally, we see that the first features to be removed by **BackSelect** are those which generally provide negative evidence against the decision.

S2.6 Understanding Differences Between Sentiment Predictors

We demonstrate how our SIS-clustering procedure can be used to understand differences in the types of concepts considered important by different neural network architectures. In addition to the LSTM (see Section S2.2), we trained a convolutional neural network (CNN) on the same sentiment analysis task (on the aroma aspect). The CNN architecture is as follows:

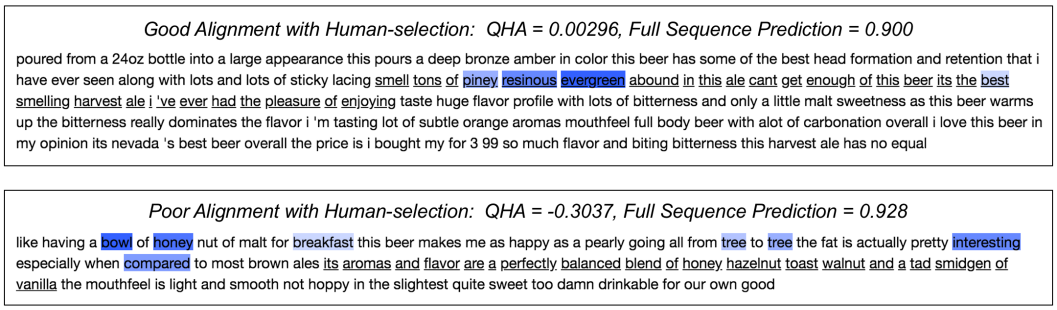


Figure S7: Beer reviews (aroma) in which human-selected sentences (underlined) are aligned well (top) and poorly (bottom) with predictive model. Fraction of SIS in the human sentences corresponds accordingly. In the bottom example (poor alignment between human-selection and predictive model), our procedure has surfaced a case where the LSTM has learned features that diverge from what a human would expect (and may suggest overfitting).

Table S3: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have positive aroma by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	smell amazing wonderful	2	nice wonderful nose	2	wonderful amazing	2	amazing amazing	2
C_2	grapefruit mango pineapple	2	pineapple grapefruit pineapple grapefruit	1	hops grapefruit pineapple floyds	1	mango pineapple incredible	1
C_3	nice smell citrus nice grapefruit taste	1	smell great complex ripe taste	1	nice smell nice hop smell pine taste	1	love nice nice smell bliss taste	1
C_4	fresh great fantastic taste	1	rich great fantastic hoped	1	fantastic cherries fantastic	1	everyone great snifters fantastic	1
C_5	awesome bounds	1	awesome grapefruit awesome	1	awesome awesome pleasing	1	awesome nailed nailed	1
C_6	creme brulee brulee	3	creme brulee decadent	1	incredible creme brulee	1	creme brulee exceptional	1
C_7	oak vanilla chocolate cinnamon vanilla oak love	1	dose oak chocolate vanilla acidic	1	vanilla figs oak thinner great	1	chocolate aroma oak vanilla dessert	1

Table S4: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have negative aroma by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	awful	15	skunky skunky	9	skunky t	7	skunky taste	6
C_2	garbage	3	taste garbage	1	garbage avoid	1	garbage rice	1
C_3	vomit	16	-	-	-	-	-	-
C_4	gross rotten	1	rotten forte	1	awkward rotten	1	rotten offputting	1
C_5	rancid horrid	1	rancid t	1	rancid	1	rancid avoid	1
C_6	rice t rice	2	rice rice	1	rice tasteless	1	budweiser rice	1

1. **Input/Embeddings Layer**: Sequence with 500 timesteps, the word at each timestep is represented by a (learned) 100-dimensional embedding
2. **Convolutional Layer 1**: Applies 128 filters of window size 3 over the sequence, with ReLU activation
3. **Max Pooling Layer 1**: Max-over-time pooling, followed by flattening, to produce a $(128,)$ representation
4. **Dense**: 1 neuron (sentiment output), sigmoid activation

Note that a new set of embeddings was learned with the CNN. As with the LSTM model, we use Adam [42] to minimize MSE on the training set. For the aroma aspect, this CNN achieves 0.016 (0.850), 0.025 (0.748), 0.026 (0.741), 0.014 (0.662) MSE (and Pearson ρ) on the Train, Validation, Test, and Annotation sets, respectively. We note that this performance is very similar to that from the LSTM (see Table S1).

We apply our procedure to extract the SIS-collection from all applicable test examples using the CNN, as in Section 4.1. Figure 9a shows the predictions from one model (LSTM or CNN) when fed input examples that are SIS extracted with respect to the *other* model (for reviews predicted to have positive sentiment toward the aroma aspect). For example, in Figure 9a, "CNN SIS Preds by LSTM" refers to predictions made by the LSTM on the set of sufficient input subsets produced by applying our **SIScollection** procedure on all examples $\mathbf{x} \in \mathcal{X}_{\text{test}}$ for which $f_{\text{CNN}}(\mathbf{x}) \geqslant \tau_+$.[4] Since the word embeddings are model-specific, we embed each SIS using the embeddings of the model making the prediction (note that while the embeddings are different, the vocabulary is the same across the models).

In Table 2, we show five example clusters (and cluster composition) resulting from clustering the combined set of all sufficient input subsets extracted by the LSTM and CNN on reviews in the test set for which a model predicts positive sentiment toward the aroma aspect. The complete clustering on reviews receiving positive sentiment predictions is shown in Table S5 and in Table S6 for reviews receiving negative sentiment predictions.

[4]For experiments involving clustering and/or comparing different models, we use examples drawn from the Test fold (instead of Annotation fold, see Table S1) to consider a larger number of examples.

Table S5: Joint clustering of the SIS extracted from beer reviews predicted to have positive aroma by LSTM or CNN model. Frequency indicates the number of occurrences of the SIS in the cluster. Percentages quantify SIS per cluster from the LSTM. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1 (LSTM: 20%)	rich chocolate	13	very rich	9	chocolate complex	5	smells rich	4
C_2 (LSTM: 21%)	great	248	amazing	119	wonderful	112	fantastic	75
C_3 (LSTM: 47%)	best smelling	23	pineapple mango	6	mango pineapple	6	pineapple grapefruit	5
C_4 (LSTM: 5%)	excellent	42	excellent flemish flemish	1	excellent excellent phenomenal	1	-	-
C_5 (LSTM: 33%)	oak chocolate	2	chocolate raisins raisins oak bourbon	1	chocolate oak	1	raisins chocolate	1
C_6 (LSTM: 5%)	goodness	19	watering goodness	1	-	-	-	-
C_7 (LSTM: 24%)	pumpkin pie	25	huge pumpkin aroma pumpkin pie	1	aroma perfect pumpkin pie taste	1	smell pumpkin nutmeg cinnamon pie	1
C_8 (LSTM: 5%)	jd	13	tremendous	8	tremendous jd	1	-	-
C_9 (LSTM: 40%)	brulee	14	creme brulee brulee	3	creme creme	1	creme brulee amazing	1
C_{10} (LSTM: 0%)	s wow	20	-	-	-	-	-	-
C_{11} (LSTM: 0%)	delicious	56	-	-	-	-	-	-
C_{12} (LSTM: 0%)	very nice	23	-	-	-	-	-	-
C_{13} (LSTM: 70%)	complex aroma	5	aroma complex peaches complex	1	aroma complex interesting cherries	1	aroma complex	1

Table S6: Joint clustering of the SIS extracted from beer reviews predicted to have negative aroma by LSTM or CNN model. Frequency indicates the number of occurrences of the SIS in the cluster. Percentages quantify SIS per cluster from the LSTM. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1 (LSTM: 29%)	not	247	no	105	bad	104	macro	94
C_2 (LSTM: 100%)	gross rotten	1	-	-	-	-	-	-
C_3 (LSTM: 100%)	rotten garbage	1	-	-	-	-	-	-
C_4 (LSTM: 62%)	vomit	26	-	-	-	-	-	-
C_5 (LSTM: 21%)	budweiser	22	sewage budweiser	1	metal budweiser	1	budweiser budweiser budweiser	1
C_6 (LSTM: 100%)	garbage rice	1	-	-	-	-	-	-
C_7 (LSTM: 3%)	n't	19	adjuncts	14	n't adjuncts	1	-	-
C_8 (LSTM: 0%)	faint	82	-	-	-	-	-	-
C_9 (LSTM: 0%)	adjunct	42	-	-	-	-	-	-

S2.7 Results for Appearance and Palate aspects

For posterity, we include results here from repeating the analysis in our paper for the two other non-aroma aspects measured in the beer reviews data: appearance and palate.

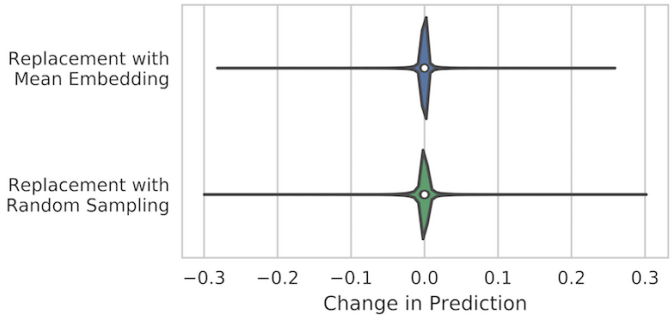


Figure S8: Change in appearance prediction ($f(\mathbf{x}\backslash\{i\}) - f(\mathbf{x})$) after masking a randomly chosen word with mean imputation or hot-deck imputation. 10,000 replacements were sampled from the appearance beer reviews training set.

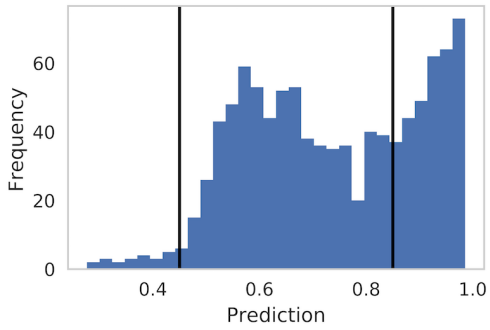


Figure S9: Predictive distribution on the annotation set (held-out) using the LSTM model for appearance. Vertical lines indicate decision thresholds ($\tau_+ = 0.85$, $\tau_- = 0.45$) selected for **SIScollection**.

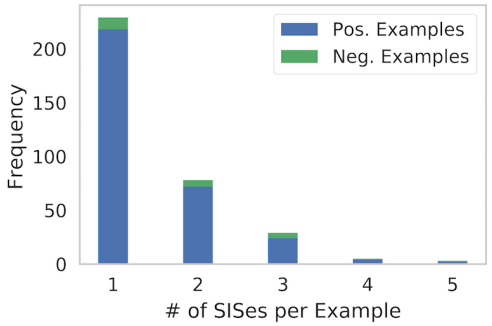


Figure S10: Number of sufficient input subsets for appearance identified by **SIScollection** per example.

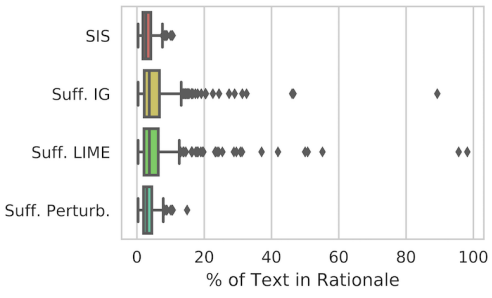


Figure S11: Length of rationales for appearance prediction

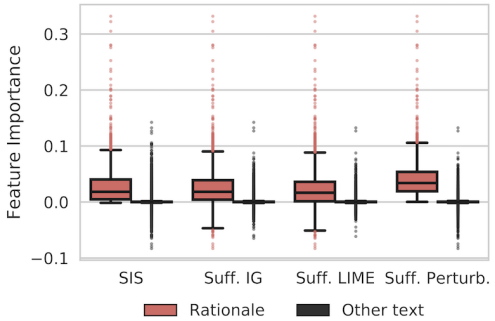


Figure S12: Importance of individual features for appearance prediction in beer review

Table S7: Statistics for rationale length and feature importance in appearance prediction. For rationale length, median and max indicate percentage of input text in the rationale. For marginal perturbed feature importance, we indicate the median importance of features in rationales and features from the other (non-rationale) text. p-values are computed using a Wilcoxon rank-sum test.

Method	Rationale Length (% of text)			Marginal Perturbed Feature Importance		
	Med.	Max	p (vs. SIS)	Med. (Rationale)	Med. (Other)	p (vs. SIS)
SIS	**2.6%**	**10.6%**	–	0.0183	1.72e-05	–
Suff. IG	3.7%	89.3%	2e-09	0.0184	2.41e-05	1e-02
Suff. LIME	3.7%	98.2%	8e-09	0.0167	2.38e-05	6e-09
Suff. Perturb.	3.0%	14.9%	9e-03	0.0339	2.51e-05	5e-44

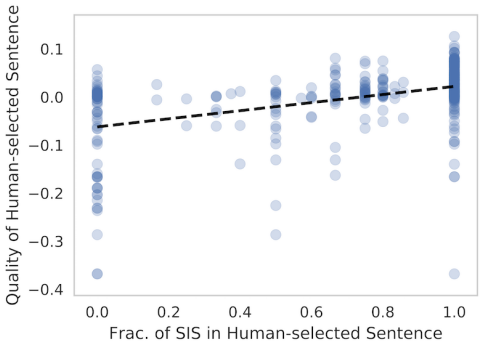


Figure S13: QHS vs. fraction of SIS in human rationale for appearance prediction

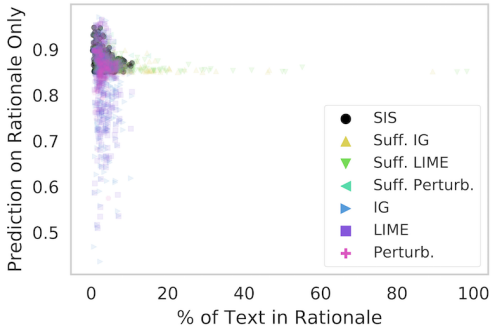


Figure S14: Prediction on rationales only vs. rationale length for various methods in positive sentiment examples for appearance. The threshold for sufficiency was $\tau_+ = 0.85$.

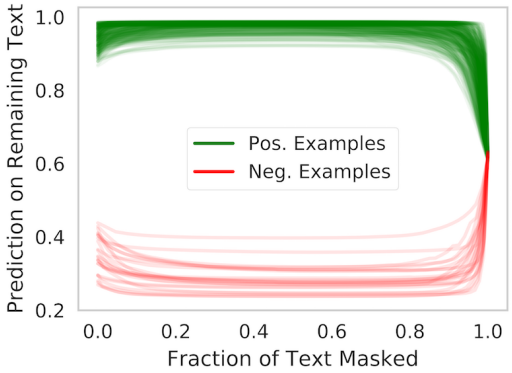


Figure S15: Prediction history on remaining (unmasked) text at each step of the **BackSelect** procedure, for examples where appearance sentiment is predicted.

Table S8: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have positive appearance by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	beautiful	376	nitro	51	looks great	38	great looking	32
C_2	gorgeous	83	-	-	-	-	-	-
C_3	beautifully	7	absolutely beautifully	2	beautifully pillowy	1	beautifully bands	1
C_4	brilliant	5	brilliant slowly	1	wonderfully brilliant	1	appearance brilliant	1
C_5	lovely looking	3	black lovely	3	impressive lovely	1	lovely crystal	1

Table S9: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have negative appearance by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	piss	46	zero	38	water water	37	water	27
C_2	unappealing	12	floaties	12	floaties unappealing	1	-	-
C_3	ugly	12	-	-	-	-	-	-

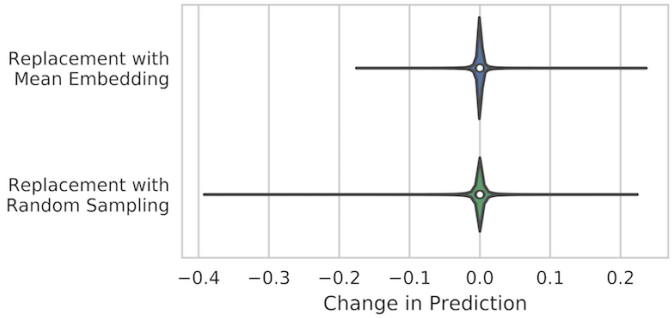


Figure S16: Change in palate prediction ($f(\mathbf{x}\backslash\{i\}) - f(\mathbf{x})$) after masking a randomly chosen word with mean imputation or hot-deck imputation. 10,000 replacements were sampled from the palate beer reviews training set.

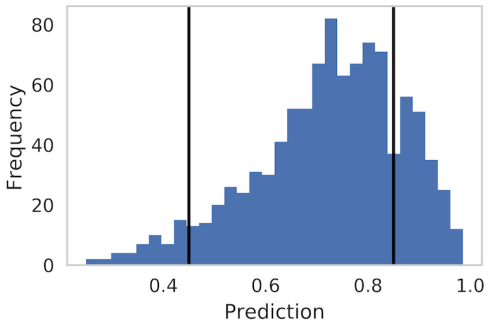


Figure S17: Predictive distribution on the annotation set (held-out) using the LSTM model for palate. Vertical lines indicate decision thresholds ($\tau_+ = 0.85$, $\tau_- = 0.45$) selected for **SIScollection**.

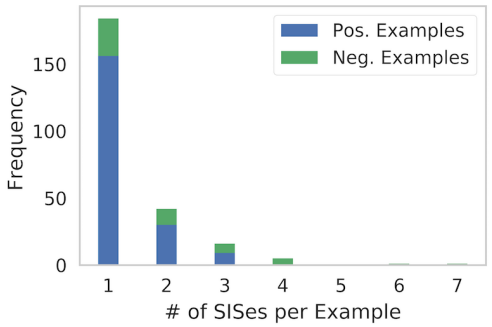


Figure S18: Number of sufficient input subsets for palate identified by **SIScollection** per example.

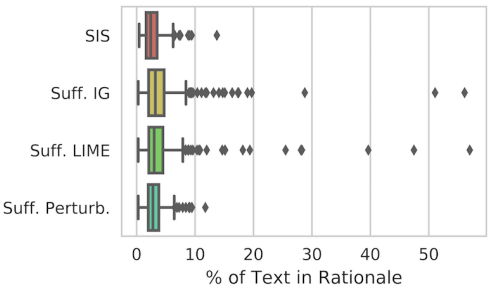


Figure S19: Length of rationales for palate prediction

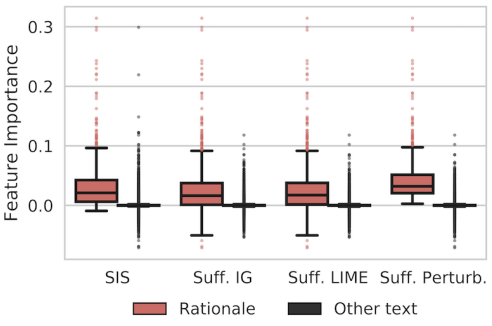


Figure S20: Importance of individual features in beer review palate rationales

Table S10: Statistics for rationale length and feature importance in palate prediction. For rationale length, median and max indicate percentage of input text in the rationale. For marginal perturbed feature importance, we indicate the median importance of features in rationales and features from the other (non-rationale) text. p-values are computed using a Wilcoxon rank-sum test.

Method	Rationale Length (% of text)			Marginal Perturbed Feature Importance		
	Med.	Max	p (vs. SIS)	Med. (Rationale)	Med. (Other)	p (vs. SIS)
SIS	**2.4%**	13.7%	–	0.0210	-8.94e-07	–
Suff. IG	3.2%	56.1%	2e-06	0.0163	-9.54e-07	6e-10
Suff. LIME	3.0%	57.0%	7e-06	0.0173	-1.19e-06	2e-07
Suff. Perturb.	2.8%	**11.8%**	3e-03	0.0319	-1.25e-06	5e-26

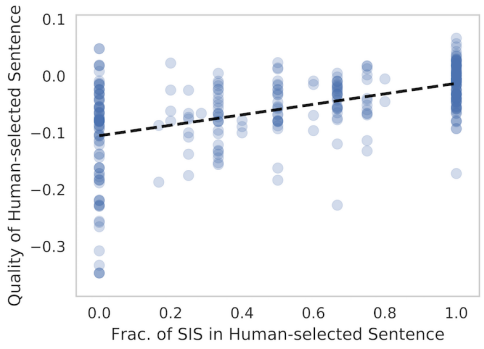


Figure S21: QHS vs. fraction of SIS in human rationale for palate prediction

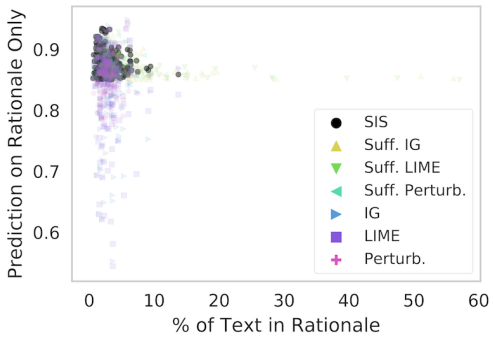


Figure S22: Prediction on rationales only vs. rationale length for various methods in positive sentiment examples for palate. The threshold for sufficiency was $\tau_+ = 0.85$.

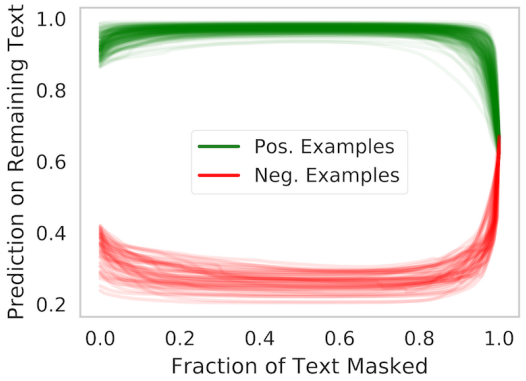


Figure S23: Prediction history on remaining (unmasked) text at each step of the **BackSelect** procedure, for examples where palate sentiment is predicted.

Table S11: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have positive palate by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster. Dashes are used in clusters with under 4 unique SIS.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	smooth creamy	27	silky smooth	20	mouthfeel perfect	16	creamy perfect	12
C_2	mouthfeel exceptional	6	exceptional mouthfeel	4	-	-	-	-
C_3	perfect	50	perfect perfect	6	-	-	-	-
C_4	smooth velvety	6	velvety smooth	6	-	-	-	-
C_5	silk	11	-	-	-	-	-	-
C_6	smooth perfect	8	mouth smooth perfect	1	perfect smooth	1	-	-
C_7	perfect great	5	great perfect	2	feels perfect	2	perfect feels great	1

Table S12: All clusters of sufficient input subsets extracted from reviews from the test set predicted to have negative palate by the LSTM. Frequency indicates the number of occurrences of the SIS in the cluster.

Cluster	SIS #1	Freq.	SIS #2	Freq.	SIS #3	Freq.	SIS #4	Freq.
C_1	overcarbonated	12	mouthfeel overcarbonated	3	way overcarbonated	1	overcarbonated mouthfeel	1
C_2	watery	302	thin	238	flat	118	mouthfeel thin	33
C_3	too carbonation masks	1	too carbonation d	1	mouthfeel odd too too	1	too carbonated admire	1
C_4	lack carbonation	4	carbonation lack	4	carbonation hurts	2	issue lack hurts	1

S3 Details of the MNIST Analysis

S3.1 Dataset and Model

The MNIST database of handwritten digits contains 60k training images and 10k test images [43]. All images are 28x28 grayscale, and we normalize them such that all pixel values are between 0 and 1. We use the convolutional architecture provided in the Keras MNIST CNN example.[5] The architecture is as follows:

1. **Input**: (28 x 28 x 1) image, all values $\in [0, 1]$
2. **Convolutional Layer 1**: Applies 32 3x3 filters with ReLU activation
3. **Convolutional Layer 2**: Applies 64 3x3 filters, with ReLU activation
4. **Pooling Layer 1**: Performs max pooling with a 2x2 filter and dropout probability 0.25
5. **Dense Layer 1**: 128 neurons, with ReLU activation and dropout probability 0.5
6. **Dense Layer 2**: 10 neurons (one per digit class), with softmax activation

The Adadelta optimizer [44] is used to minimize cross-entropy loss on the training set. The final model achieves 99.7% accuracy on the train set and 99.1% accuracy on the held-out test set.

S3.2 Local Minima in Backward Selection

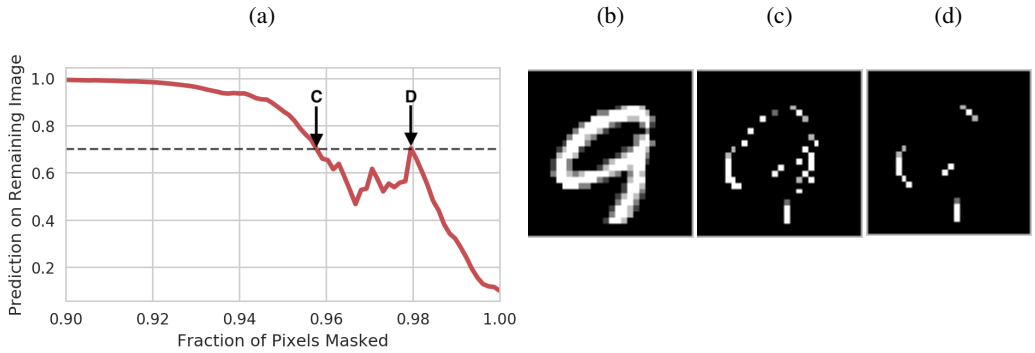


Figure S24: **(a)** Prediction on remaining image as pixels are masked during backward selection, when our CNN classifier is fed the MNIST digit in (b). The dashed line depicts the threshold $\tau = 0.7$. **(b)** Original image (class 9). **(c)** SIS if backward selection were to terminate the first time prediction on remaining image drops below 0.7, corresponding to point **C** in (a) (CNN predicts class 9 with probability 0.700 on this SIS). **(d)** Actual SIS produced by our **FindSIS** algorithm, corresponding to point **D** in (a) (CNN predicts class 9 with probability 0.704 on this SIS).

Figure S24 demonstrates an example MNIST digit for which there exists a local minimum in the backward selection phase of our algorithm to identify the initial SIS. Note that if we were to terminate the backward selection as soon as predictions drop below the decision threshold, the resulting SIS would be overly large, violating our minimality criterion. It is also evident from Figure S24 that the smaller-cardinality SIS in (d), found after the initial local optimum in (c), presents a more interpretable input pattern that enables better understanding of the core motifs influencing our classifier's decisions. To avoid suboptimal results, it is important to run a complete backward selection sweep until the entire input is masked before building the SIS upward, as done in our **SIScollection** procedure.

S3.3 Energy Distance Between Image SIS

To cluster SIS from the image data, we compute the pairwise distance between two SIS subsets S_1 and S_2 as the energy distance [45] between two distributions over the image pixel coordinates that

[5] `http://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py`

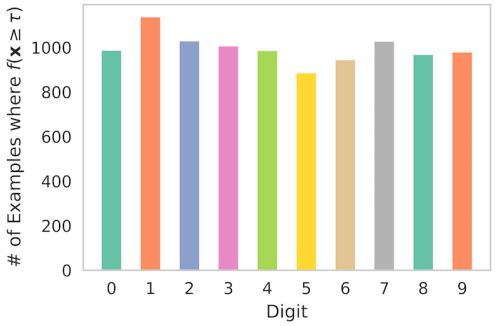


Figure S25: Number of examples per digit in the test set for which $f(\mathbf{x}) \geqslant \tau$ for the top class. The complete set of sufficient input subsets is computed for all of these examples.

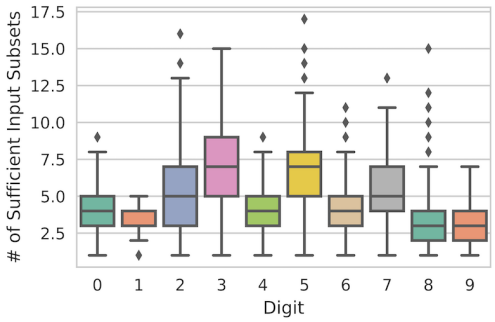


Figure S26: Distributions of number of sufficient input subsets identified per image, by digit.

comprise the SIS, X_1 and $X_2 \in \mathbb{R}^2$:

$$D(X_1, X_2) = 2 \cdot \mathbb{E}\,||X_1 - X_2|| - \mathbb{E}\,||X_1 - X_1'|| - \mathbb{E}\,||X_2 - X_2'|| \geqslant 0$$

Here, X_i is uniformly distributed over the pixels that are selected as part of the SIS subset S_i, X_i' is an i.i.d. copy of X_i, and $||\cdot||$ represents the Euclidean norm. Unlike a Euclidean distance between images, our usage of the energy distance takes into account distances between the similar pixel coordinates that comprise each SIS. The energy distance offers a more efficiently computable integral probability metric than the optimal transport distance, which has been widely adopted as an appropriate measure of distance between images.

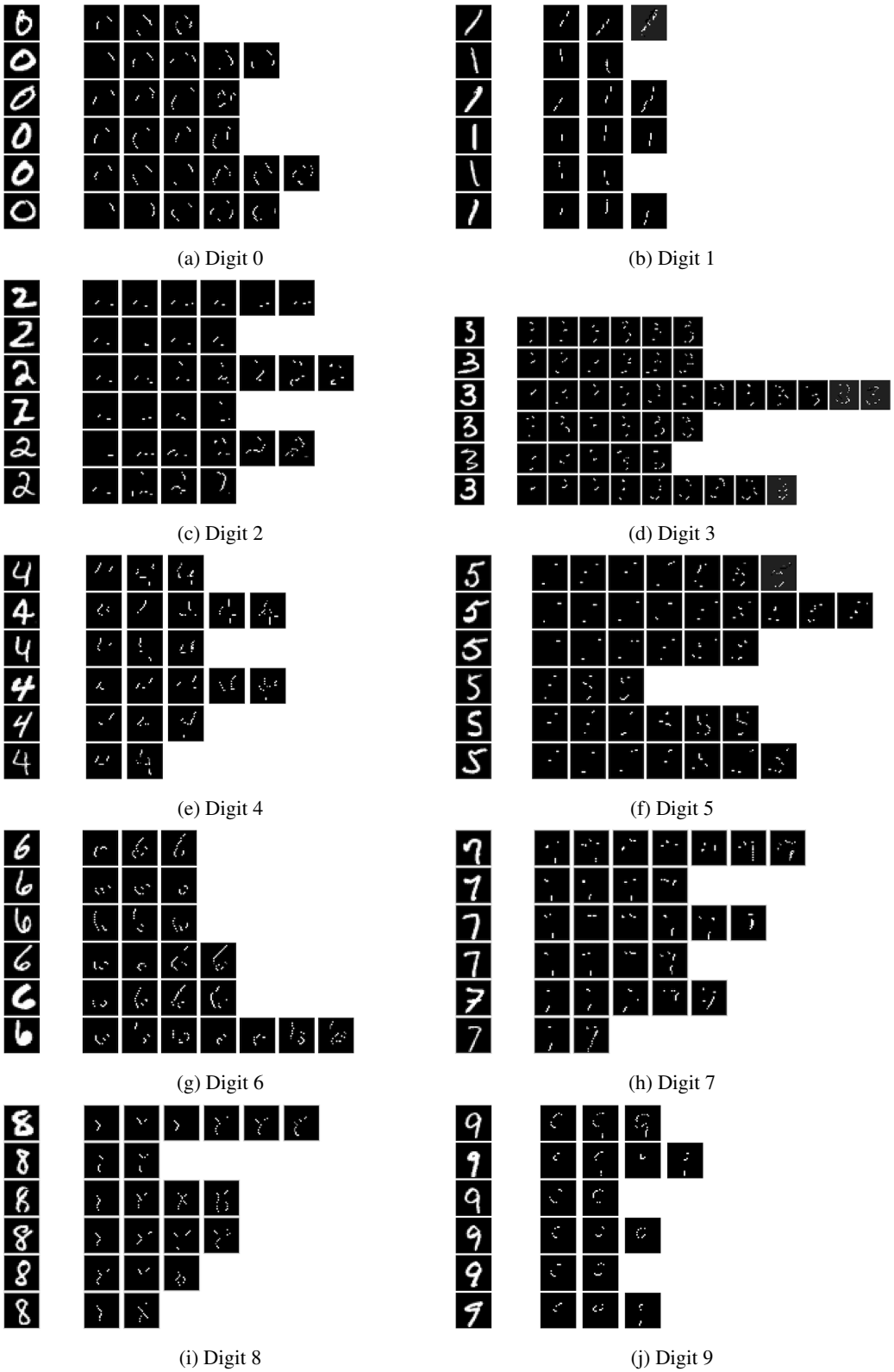
S3.4 SIS Clustering and Adversarial Analysis

We set the threshold $\tau = 0.7$ for SIS to ensure that the model is confident in its class prediction (probability of the predicted class is $\geqslant 0.7$). Almost all test examples initially have $f(\mathbf{x}) \geqslant \tau$ for the top class (Figure S25). We identify all test examples that satisfy this condition and use SIS to identify all sufficient input subsets. The number of sufficient input subsets per digit is shown in Figure S26.

We apply our **SIScollection** algorithm to identify sufficient input subsets on MNIST test digits (Section 4.2). Examples of the complete SIS-collection corresponding to randomly chosen digits are shown in Figure S27. We also cluster all the sufficient input subsets identified for each class (Section 4.3), depicting the results in Figure S28.

In Figure 6, we show an MNIST image of the digit 9, adversarially perturbed to 4, and the sufficient subsets corresponding to the adversarial prediction. Although a visual inspection of the perturbed image does not really reveal exactly how it has been manipulated, it becomes immediately clear from the SIS-collection for the adversarial image. These sets shows that the perturbation modifies pixels in such a way that input patterns similar to the typical SIS-collection for a 4 (Figure 5) become embedded in the image. The adversarial manipulation was done using the Carlini-Wagner L_2 (CW2) attack[6] [47] with a confidence parameter of 10. The CW2 attack tries to find the minimal change to the image, with respect to the L_2 norm, that will lead the image to be misclassified. It has been demonstrated to be one of the strongest extant adversarial attacks [48].

[6]Implemented in the cleverhans library [46]



(a) Digit 0

(b) Digit 1

(c) Digit 2

(d) Digit 3

(e) Digit 4

(f) Digit 5

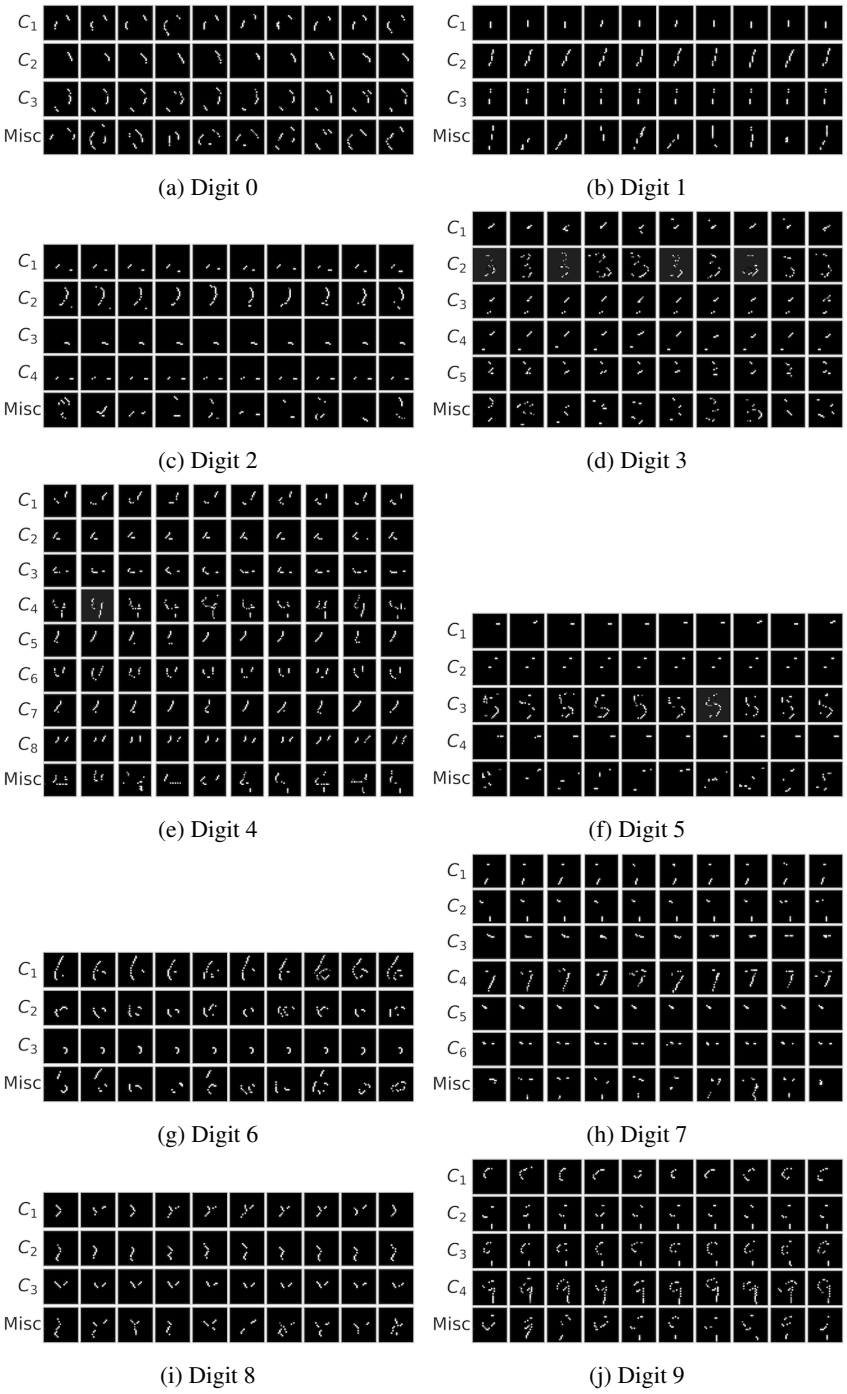
(g) Digit 6

(h) Digit 7

(i) Digit 8

(j) Digit 9

Figure S27: Visualization of SIS-collections identified from MNIST digits that are confidently classified by the CNN. For each class, six examples were chosen randomly. For each example, we show the original image (left) and the complete set of sufficient input subsets identified for that example (remaining images in each row). Each individual SIS satisfies $f(\mathbf{x}_S) \geqslant \tau$ for that class.



(a) Digit 0 (b) Digit 1

(c) Digit 2 (d) Digit 3

(e) Digit 4 (f) Digit 5

(g) Digit 6 (h) Digit 7

(i) Digit 8 (j) Digit 9

Figure S28: Clustering all the SIS found for each digit under the CNN model (see Section 4.3). Each row contains images drawn from one cluster. The bottom row ("Misc") contains a sample of miscellaneous SIS not assigned to any cluster by DBSCAN.

S3.5 Understanding Differences Between MNIST Classifiers

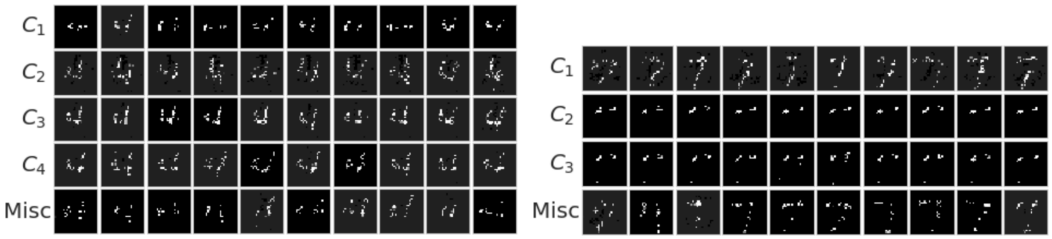
We use SIS and our clustering procedure to understand and visualize differences in features learned by two different models trained on the same MNIST digit classification task. In addition to the previously-described CNN model (see Section S3.1), we also trained a simple multilayer perceptron (MLP) on the same task. The MLP architecture is as follows:

1. **Input**: 784-dimensional (flattened) image, all values $\in [0, 1]$
2. **Dense Layer 1**: 250 neurons, ReLU activation, and dropout probability 0.2
3. **Dense Layer 2**: 250 neurons, ReLU activation, and dropout probability 0.2
4. **Dense Layer 3**: 10 neurons (one per digit class), with softmax activation

As with the CNN, Adadelta [44] is used to minimize cross-entropy loss on the training set. The final MLP model achieves 99.7% accuracy on the train set and 98.3% accuracy on the test set, which is close to the performance of the CNN (see Section S3.1).

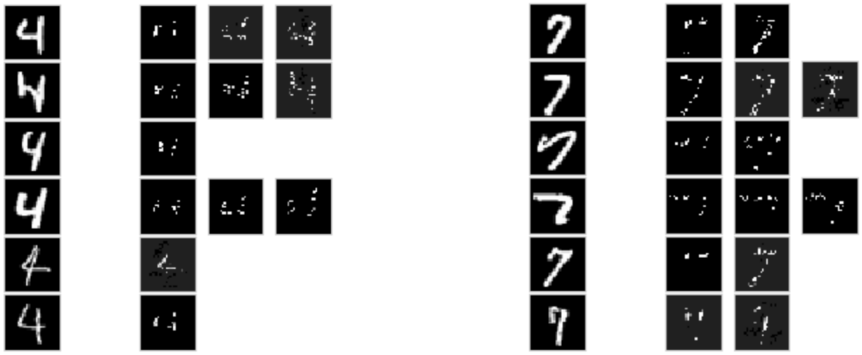
We apply the same procedure as in Section 4.2 to extract the SIS-collection from all applicable test images using the MLP. To understand differences between the feature patterns that each model has learned to associate with predicting each digit, we combine all SIS (from both models for a particular class) and run our clustering procedure (see Section 4.3 and Figure 8). In the resulting clustering, we list what percentage of the SIS in each cluster stem from the CNN vs. the MLP. Most clusters contain examples purely from a single model, indicating the two models have learned to associate different feature patterns with the target class (Figure 8), which was chosen to be the digit 4 in this case.

For further comparison, we include clustering results for the SIS extracted from the MLP as evidence for digits 4 and 7 (Figure S29). Additionally, Figure S30 shows all of the SIS extracted from example digits from these classes applying our procedure on the MLP.



(a) Digit 4 (b) Digit 7

Figure S29: Clustering all the SIS identified by our method on digits 4 and 7 under the MLP model (see Section 4.3). Each row contains images drawn from one cluster. The bottom row ("Misc") contains a sample of miscellaneous SIS not assigned to any cluster by DBSCAN. Compare to the SIS-clustering from our CNN model (Figure S28).



(a) Digit 4 (b) Digit 7

Figure S30: Visualization of SIS-collections identified for MNIST digits 4 and 7 under the MLP model. For each class, six examples were chosen randomly. For each example, we show the original image (left) and the complete set of sufficient input subsets identified for that example (remaining images in each row). Note that each individual SIS satisfies $f(\mathbf{x}_S) \geqslant \tau$ for that class. Compare to the SIS extracted from our CNN (Figure S27).

Additional References for the Supplementary Information

[36] Sundararajan M, Taly A, Yan Q (2017) Axiomatic attribution for deep networks. In: *International Conference on Machine Learning*.

[37] Ribeiro MT, Singh S, Guestrin C (2016) "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*.

[38] Lei T, Barzilay R, Jaakkola T (2016) Rationalizing neural predictions. In: *Empirical Methods in Natural Language Processing*.

[39] McAuley J, Leskovec J, Jurafsky D (2012) Learning attitudes and attributes from multi-aspect reviews. In: *IEEE 12th International Conference on Data Mining*. pp. 1020–1025.

[40] Wang Y, Huang M, Zhao L, et al. (2016) Attention-based lstm for aspect-level sentiment classification. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

[41] Radford A, Jozefowicz R, Sutskever I (2017) Learning to generate reviews and discovering sentiment. *arXiv:170401444* .

[42] Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: *International Conference on Learning Representations*.

[43] LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86: 2278–2324.

[44] Zeiler MD (2012) Adadelta: An adaptive learning rate method. *arXiv:12125701* .

[45] Rizzo ML, Székely GJ (2016) Energy distance. *Wiley Interdisciplinary Reviews: Computational Statistics* 8: 27–38.

[46] Papernot N, Carlini N, Goodfellow I, Feinman R, Faghri F, et al. (2017) cleverhans v2.0.0: an adversarial machine learning library. *arXiv:161000768* .

[47] Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. In: *IEEE Symposium on Security and Privacy*.

[48] Carlini N, Wagner D (2017) Adversarial examples are not easily detected: Bypassing ten detection methods. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*.