On Guaranteed Optimal Robust Explanations for NLP Models

Emanuele La Malfa[1*], **Agnieszka Zbrzezny**[1,2], **Rhiannon Michelmore**[1]
Nicola Paoletti[3] and **Marta Kwiatkowska**[1]

[1]University of Oxford, [2]University of Warmia and Mazury in Olsztyn,
[3]Royal Holloway, University of London

Abstract

We build on abduction-based explanations for machine learning and develop a method for computing local explanations for neural network models in natural language processing (NLP). Our explanations comprise a subset of the words of the input text that satisfies two key features: optimality w.r.t. a user-defined cost function, such as the length of explanation, and robustness, in that they ensure prediction invariance for any bounded perturbation in the embedding space of the left-out words. We present two solution algorithms, respectively based on implicit hitting sets and maximum universal subsets, introducing a number of algorithmic improvements to speed up convergence of hard instances. We show how our method can be configured with different perturbation sets in the embedded space and used to detect bias in predictions by enforcing include/exclude constraints on biased terms, as well as to enhance existing heuristic-based NLP explanation frameworks such as Anchors. We evaluate our framework on three widely used sentiment analysis tasks and texts of up to 100 words from SST, Twitter and IMDB datasets, demonstrating the effectiveness of the derived explanations[1].

1 Introduction

The increasing prevalence of deep learning models in real-world decision-making systems has made AI explainability a central problem, as we seek to complement such highly-accurate but opaque models with comprehensible explanations as to why the model produced a particular prediction [Samek and others, 2017; Ribeiro and others, 2016; Zhang and others, 2019; Liu and others, 2018; Letham and others, 2015]. Amongst existing techniques, *local explanations* explain the individual prediction in terms of a subset of the input features that justify the prediction. State-of-the-art explainers such as LIME and Anchors [Ribeiro and others, 2016; Ribeiro and others, 2018] use heuristics to obtain short explanations, which may generalise better beyond the given input and are more easily interpretable to human experts, but lack robustness to adversarial perturbations. The abduction-based method of [Ignatiev and others, 2019a], on the other hand, ensures minimality and robustness of the prediction by requiring its invariance w.r.t. any perturbation of the left-out features, meaning that the explanation is sufficient to imply the prediction. However, since perturbations are potentially unbounded, this notion of robustness may not be appropriate for certain applications.

In this paper, we focus on natural language processing (NLP) neural network models and, working in the embedding space with words as features, introduce *optimal robust explanations (OREs)*. OREs are *provably guaranteed* to be both *robust*, in the sense that the prediction is invariant for any (reasonable) replacement of the features outside the explanation, and *minimal* for a given user defined cost function, such as the length of the explanation. Our core idea shares similarities with abduction-based explanations (ABE) of [Ignatiev and others, 2019a], but is better suited to NLP models, where the unbounded nature of ABE perturbations may result in trivial explanations equal to the entire input. We show that OREs can be formulated as a particular kind of ABE or, equivalently, minimal satisfying assignment (MSA). We develop two methods to compute OREs by extending existing algorithms for ABEs and MSAs [Ignatiev and others, 2019a; Dillig and others, 2012]. In particular, we incorporate state-of-the-art robustness verification methods [Katz and others, 2019; Wang and others, 2018] to solve entailment/robustness queries and improve convergence by including sparse adversarial attacks and search tree reductions. By adding suitable constraints, we show that our approach allows one to detect biased decisions [Darwiche and Hirth, 2020] and enhance heuristic explainers with robustness guarantees [Ignatiev and others, 2019d].

To the best of our knowledge, this is the first method to derive local explanations for NLP models with provable robustness and optimality guarantees. We empirically demonstrate that our approach can provide useful explanations for non-trivial fully-connected and convolutional networks on three widely used sentiment analysis benchmarks (SST, Twitter and IMDB). We compare OREs with the popular Anchors method, showing that Anchors often lack prediction robustness in our benchmarks, and demonstrate the usefulness of

*First Author, contact at emanuele.la.malfa@cs.ox.ac.uk

[1]Code available at https://github.com/EmanueleLM/OREs

our framework on model debugging, bias evaluation, and repair of non-formal explainers like Anchors.

2 Related Work

Interpretability of machine learning models is receiving increasing attention [Chakraborty and others, 2017]. Existing methods broadly fall in two categories: explanations via globally interpretable models (e.g. [Wang and Rudin, 2015; Zhang and others, 2018]), and local explanations for a given input and prediction (to which our work belongs). Two prominent examples of the latter category are LIME [Ribeiro and others, 2016], which learns a linear model around the neighbourhood of an input using random local perturbations, and Anchors [Ribeiro and others, 2018] (introduced in Section 3). These methods, however, do not consider robustness, making them fragile to adversarial attacks and thus insufficient to imply the prediction. Repair of non-formal explainers has been studied in [Ignatiev and others, 2019d] but only for boosted trees predictors. [Narodytska and others, 2019] assesses the quality of Anchors' explanations by encoding the model and explanation as a propositional formula. The explanation quality is then determined using model counting, but for binarised neural networks only. Other works that focus on binarised neural networks, Boolean classifiers or similar representations include [Shi and others, 2020; Darwiche and Hirth, 2020; Darwiche, 2020]. Methods tailored to (locally) explaining NLP model decisions for a given input include [Li and others, 2015; Singh and others, 2018]. These identify input features, or clusters of input features, that most contribute to the prediction, using saliency and agglomerative contextual decomposition respectively. Layer-wise relevance propagation [Bach and others, 2015] is also popular for NLP explanations, and is used in [Arras and others, 2016; Arras and others, 2017; Ding and others, 2017]. Similarly to the above, these methods do not consider robustness. Robustness of neural network NLP models to adversarial examples has been studied in [Huang and others, 2019; Jia and others, 2019; La Malfa and others, 2020]. We note that robustness verification is a different (and arguably simpler) problem from deriving a robust explanation, as the latter requires performing multiple robustness verification queries (see Section 4). Existing neural network verification approaches include symbolic (SMT) [Katz and others, 2019], relaxation [Ko and others, 2019; Wang and others, 2018], and global optimisation [Ruan and others, 2018]. Research utilising hitting sets can be seen in [Ignatiev and others, 2019c], which relates explanations and adversarial examples through a generalised form of hitting set duality, and [Ignatiev and others, 2019b], which works on improving model-based diagnoses by using an algorithm based on hitting sets to filter out non-subset-minimal sets of diagnoses.

3 Optimal Robust Explanations for NLP

Preliminaries We consider a standard NLP classification task where we classify some given input text t into a plausible class y from a finite set $\mathcal{Y}$. We assume that t is a fixed length sequence of words (i.e., *features*) l, $t = (w_1, \ldots, w_l)$, where $w_i \in W$ with W being a finite vocabulary (possibly including padding). Text inputs are encoded using a continuous *word embedding* $\mathcal{E} : W \to \mathbb{R}^d$, where d is the size of the embedding [Mikolov and others, 2013]. Thus, given a text $t = (w_1, \ldots, w_l)$, we define the embedding $\mathcal{E}(t)$ of t as the sequence $x = (x_{w_1}, \ldots, x_{w_l}) \in \mathbb{R}^{l \cdot d}$, where $x_{w_i} = \mathcal{E}(w_i)$. We denote with $W_{\mathcal{E}} \subseteq W$ the vocabulary used to train $\mathcal{E}$. We consider embedding vectors trained from scratch on the sentiment task, a technique that enforces words that are positively correlated to each of the output classes to be gathered closer in the embedding space [Baroni and others, 2014], which is considered a good proxy for semantic similarity with respect to the target task compared to count-based embeddings [Alzantot and others, 2018]. For classification we consider a *neural network* $M : \mathbb{R}^{l \cdot d} \to \mathcal{Y}$ that operates on the text embedding.

Robust Explanations In this paper, we seek to provide *local explanations* for the predictions of a neural network NLP model. For a text embedding $x = \mathcal{E}(t)$ and a prediction $M(x)$, a local explanation E is a subset of the features of t, i.e., $E \subseteq F$ where $F = \{w_1, \ldots, w_l\}$, that is sufficient to imply the prediction. We focus on deriving *robust explanations*, i.e., on extracting a subset E of the text features F which ensure that the neural network prediction remains invariant for any perturbation of the other features $F \setminus E$. Thus, the features in a robust explanation are *sufficient to imply the prediction* that we aim to explain, a clearly desirable feature for a local explanation. In particular, we focus on explanations that are *robust w.r.t. bounded perturbations in the embedding space of the input text*. We extract word-level explanations by means of word embeddings: we note that OREs work, without further extensions, with diverse representations (e.g., sentence-level, characters-level, etc.). For a word $w \in W$, with embedding $x_w = \mathcal{E}(w)$ we denote with $\mathcal{B}(w) \subseteq \mathbb{R}^d$ a generic set of word-level perturbations. We consider the following kinds of perturbation sets, depicted also in Fig. 1.

ϵ-ball: $\mathcal{B}(w) = \{x \in \mathbb{R}^d \mid \|x - x_w\|_p \leq \epsilon\}$, for some $\epsilon > 0$ and $p > 0$. This is a standard measure of local robustness in computer vision, where ϵ-variations are interpreted as manipulations of the pixel intensity of an image. It has also been adopted in early NLP robustness works [Miyato and others, 2016], but then replaced with better representations based on actual word replacements and their embeddings, see below.

k-NN box closure: $\mathcal{B}(w) = BB(\mathcal{E}(NN_k(w)))$, where $BB(X)$ is the minimum bounding box for set X; for a set $W' \subseteq W$, $\mathcal{E}(W') = \bigcup_{w' \in W'} \{\mathcal{E}(w')\}$; and $NN_k(w)$ is the set of the k closest words to w in the embedding space, i.e., words w' with smallest $d(x_w, \mathcal{E}(w'))$, where d is a valid notion of distance between embedded vectors, such as p-norms or cosine similarity[2]. This provides an over-approximation of the k-NN convex closure, for which constraint propagation (and thus robustness checking) is more efficient [Jia and others, 2019; Huang and others, 2019].

For some word-level perturbation $\mathcal{B}$, set of features $E \subseteq F$, and input text t with embedding $(x_1, \ldots, x_l)$, we denote

[2]even though the box closure can be calculated for any set of embedded words.

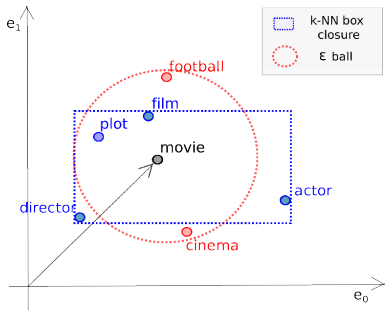

Figure 1: A graphical representation of the perturbation sets we define in the embedding space.

with $\mathcal{B}_E(t)$ the set of *text-level* perturbations obtained from t by keeping constant the features in E and perturbing the others according to $\mathcal{B}$:

$$\mathcal{B}_E(t) = \{(x'_1, \ldots, x'_l) \in \mathbb{R}^{l \cdot d} \mid x'_w = x_w \text{ if } w \in E; \\ x'_w \in \mathcal{B}(w) \text{ otherwise}\}. \quad (1)$$

A robust explanation $E \subseteq F$ ensures prediction invariance for any point in $\mathcal{B}_E(t)$, i.e., any perturbation (within $\mathcal{B}$) of the features in $F \setminus E$.

Def. 1 (Robust Explanation)**.** *For a text $t = (w_1, \ldots, w_l)$ with embedding $x = \mathcal{E}(t)$, word-level perturbation $\mathcal{B}$, and classifier M, a subset $E \subseteq F$ of the features of t is a* robust explanation *iff*

$$\forall x' \in \mathcal{B}_E(t).\ M(x') = M(x). \quad (2)$$

We denote (2) *with predicate* $\mathsf{Rob}_{M,x}(E)$.

Optimal Robust Explanations (OREs) While robustness is a desirable property, it is not enough alone to produce useful explanations. Indeed, we can see that an explanation E including all the features, i.e., $E = F$, trivially satisfies Definition 1. Typically, one seeks short explanations, because these can generalise to several instances beyond the input x and are easier for human decision makers to interpret. We thus introduce *optimal robust explanations (OREs)*, that is, explanations that are both robust and optimal w.r.t. an arbitrary cost function that assign a penalty to each word.

Def. 2 (Optimal Robust Explanation)**.** *Given a cost function $\mathcal{C} : W \to \mathbb{R}^+$, and for $t = (w_1, \ldots, w_l)$, x, $\mathcal{B}$, and M as in Def. 1, a subset $E^* \subseteq F$ of the features of t is an* ORE *iff*

$$E^* \in \arg\min_{E \subseteq F} \sum_{w \in E} \mathcal{C}(w) \text{ s.t. } \mathsf{Rob}_{M,x}(E). \quad (3)$$

Note that (3) is always feasible, because its feasible set always includes at least the trivial explanation $E = F$. A special case of our OREs is when $\mathcal{C}$ is *uniform* (it assigns the same cost to all words in t), in which case E^* is (one of) the *robust explanations of smallest size*, i.e., with the least number of words.

Relation with Abductive Explanations Our OREs have similarities with the *abduction-based explanations (ABEs)* of [Ignatiev and others, 2019a] in that they also derive minimal-cost explanations with robustness guarantees. For an input text $t = (w_1, \ldots, w_l)$, let $C = \bigwedge_{i=1}^{l} \chi_i = x_{w_i}$ be the *cube* representing the embedding of t, where χ_i is a variable denoting the i-th feature of x. Let $\mathcal{N}$ represent the logical encoding of the classifier M, and $\hat{y}$ be the formula representing the output of $\mathcal{N}$ given $\chi_1, \ldots, \chi_l$.

Def. 3 ([Ignatiev and others, 2019a])**.** *An* abduction-based explanation (ABE) *is a minimal cost subset C^* of C such that $C^* \wedge \mathcal{N} \models \hat{y}$.*

Note that the above entailment is equivalently expressed as $C^* \models (\mathcal{N} \to \hat{y})$. Let $B = \bigwedge_{i=1}^{l} \chi_i \in \mathcal{B}(w_i)$ be the constraints encoding our perturbation space. Then, the following proposition shows that OREs can be defined in a similar abductive fashion and also in terms of *minimum satisfying assignments (MSAs)* [Dillig and others, 2012]. In this way, we can derive OREs via analogous algorithms to those used for ABEs [2019a] and MSAs [Dillig and others, 2012], as explained in Section 4. Moreover, we find that every ORE can be formulated as a prime implicant [Ignatiev and others, 2019a], a property that connects our OREs with the notion of sufficient reason introduced in [Darwiche and Hirth, 2020].

Prop. 1. *Let E^* be an ORE and C^* its constraint encoding. Define $\phi \equiv (B \wedge \mathcal{N}) \to \hat{y}$. Then, all the following definitions apply to C^*:*

1. *C^* is a minimal cost subset of C such that $C^* \models \phi$.*
2. *C^* is a minimum satisfying assignment for ϕ.*
3. *C^* is a prime implicant of ϕ.*

Proof. See supplement □

The key difference with ABEs is that our OREs are robust to *bounded* perturbations of the excluded features, while ABEs must be robust to *any* possible perturbation. This is an important difference because it is hard (often impossible) to guarantee prediction invariance w.r.t. the entire input space when this space is continuous and high-dimensional, like in our NLP embeddings. In other words, if for our NLP tasks we allowed any word-level perturbation as in ABEs, in most cases the resulting OREs will be of the trivial kind, $E^* = F$ (or $C^* = C$), and thus of little use. For example, if we consider ϵ-ball perturbations and the review *"the gorgeously elaborate continuation of the lord of the rings"*, the resulting smallest-size explanation is of the trivial kind (it contains the whole review) already at $\epsilon = 0.1$.

Exclude and include constraints We further consider OREs E^* derived under constraints that enforce specific features F' to be included/excluded from the explanation:

$$E^* \in \arg\min_{E \subseteq F} \sum_{w \in E} \mathcal{C}(w) \text{ s.t. } \mathsf{Rob}_{M,x}(E) \wedge \phi(E), \quad (4)$$

where $\phi(E)$ is one of $F' \cap E = \emptyset$ (*exclude*) and $F' \subseteq E$ (*include*). Note that adding *include* constraints doesn't affect the feasibility of our problem[3]. Conversely, *exclude* constraints

[3]because the feasible region of (4) always contains at least the explanation $E^* \cup F'$, where E^* is a solution of (3) and F' are the features to include. See Def. 1.

might make the problem infeasible when the features in F' don't admit perturbations, i.e., they are necessary for the prediction, and thus cannot be excluded. Such constraints can be easily accommodated by any solution algorithm for non-constrained OREs: for *include* ones, it is sufficient to restrict the feasible set of explanations to the supersets of F'; for *exclude* constraints, we can manipulate the cost function so as to make any explanation with features in F' strictly sub-optimal w.r.t. explanations without[4].

Constrained OREs enable two crucial use cases: *detecting biased decisions*, and *enhancing non-formal explainability frameworks*.

Detecting bias Following [Darwiche and Hirth, 2020], we deem a classifier decision *biased* if it depends on protected features, i.e., a set of input words that should not affect the decision (e.g., a movie review affected by the director's name). In particular, a decision $M(x)$ is biased if we can find, within a given set of text-level perturbations, an input x' that agrees with x on all but protected features and such that $M(x) \neq M(x')$.

Def. 4. *For classifier M, text t with features F, protected features F' and embedding $x = \mathcal{E}(t)$, decision $M(x)$ is* biased *w.r.t. some word-level perturbation $\mathcal{B}$, if*

$$\exists x' \in \mathcal{B}_{F \setminus F'}(t). M(x) \neq M(x').$$

The proposition below allows us to use exclude constraints to detect bias.

Prop. 2. *For M, t, F, F', x and $\mathcal{B}$ as per Def. 4,* decision $M(x)$ is biased iff (4) is infeasible under $F' \cap E = \emptyset$.

Proof. See supplement □

Enhancing non-formal explainers The local explanations produced by heuristic approaches like LIME or Anchors do not enjoy the same robustness/invariance guarantees of our OREs. We can use our approach to *minimally extend* (w.r.t. the chosen cost function) any non-robust local explanation F' in order to make it robust, by solving (4) under the *include* constraint $F' \subseteq E$. In particular, with a uniform $\mathcal{C}$, our approach would identify the smallest set of extra words that make F' robust. Being minimal/smallest, such an extension retains to a large extent the original explainability properties.

Relation with Anchors Anchors [Ribeiro and others, 2018] are a state-of-the-art method for ML explanations. Given a perturbation distribution $\mathcal{D}$, classifier M and input x, an anchor A is a predicate over the input features such that $A(x)$ holds and A has high *precision* and *coverage*, defined next.

$$\mathsf{prec}(A) = \Pr_{\mathcal{D}(x' \mid A(x'))}(M(x) = M(x')); \; \mathsf{cov}(A) = \Pr_{\mathcal{D}(x')}(A(x')) \tag{5}$$

In other words, $\mathsf{prec}(A)$ is the probability that the prediction is invariant for any perturbation x' to which explanation A applies. In this sense, precision can be intended as a robustness probability. $\mathsf{cov}(A)$ is the probability that explanation A applies to a perturbation. To discuss the relation between Anchors and OREs, for an input text t, consider an arbitrary distribution $\mathcal{D}$ with support in $\mathcal{B}_\emptyset(t)$ (the set of all possible text-level perturbations), see (1); and consider anchors A defined as subsets E of the input features F, i.e., $A_E(x) = \bigwedge_{w \in E} x_w = \mathcal{E}(w)$. Then, our OREs enjoy the following properties.

Prop. 3. *If E is a robust explanation, then* $\mathsf{prec}(A_E) = 1$.

Proof. See supplement □

Note that when $\mathcal{D}$ is continuous, $\mathsf{cov}(A_E)$ is always zero unless $E = \emptyset$[5]. We thus illustrate the next property assuming that $\mathcal{D}$ is discrete (when $\mathcal{D}$ is continuous, the following still applies to any empirical approximation of $\mathcal{D}$).

Prop. 4. *If $E \subseteq E'$, then* $\mathsf{cov}(A_E) \geq \mathsf{cov}(A_{E'})$.

Proof. See supplement □

The above proposition suggests that using a uniform $\mathcal{C}$, i.e., minimizing the explanation's length, is a sensible strategy to obtain high-coverage OREs.

4 Solution Algorithms

We present two solution algorithms to derive OREs, respectively based on the hitting-set (HS) paradigm of [Ignatiev and others, 2019a] and the MSA algorithm of [Dillig and others, 2012]. Albeit different, both algorithms rely on repeated entailment/robustness checks $B \wedge E \wedge \mathcal{N} \models \hat{y}$ for a candidate explanation $E \subset C$. For this check, we employ two state-of-the-art neural network verification tools, Marabou [Katz and others, 2019] and Neurify [Wang and others, 2018]: they both give provably correct answers and, when the entailment is not satisfied, produce a counter-example $x' \in \mathcal{B}_E(t)$, i.e., a perturbation that agrees with E and such that $B \wedge C' \wedge \mathcal{N} \not\models \hat{y}$, where C' is the cube representing x'. We now briefly outline the two algorithms. A more detailed discussion (including the pseudo-code) is available in the supplement.

Minimum Hitting Set For a counterexample C', let I' be the set of feature variables where C' does not agree with C (the cube representing the input). Then, every explanation E that satisfies the entailment must hit all such sets I' built for any counter-examples C' [Ignatiev and others, 2016]. Thus, the HS paradigm iteratively checks candidates E built by selecting the subset of C whose variables form a minimum HS (w.r.t. cost $\mathcal{C}$) of said I's. However, we found that this method often struggles to converge for our NLP models, especially with large perturbations spaces (i.e., large ϵ or k). We solved this problem by extending the HS approach with a sub-routine that generates batches of *sparse adversarial attacks* for the input C. This has a two-fold benefit: 1) we reduce the number of entailment queries required to produce counter-examples,

[4] That is, we use cost $\mathcal{C}'$ such that $\forall_{w \in F \setminus F'} \mathcal{C}'(w) = \mathcal{C}(w)$ and $\forall_{w' \in F'} \mathcal{C}'(w') > \sum_{w \in F \setminus F'} \mathcal{C}(w)$. The ORE obtained under cost $\mathcal{C}'$ might still include features from F', which implies that (4) is infeasible (i.e., no robust explanation without elements of F' exists).

[5] in which case $\mathsf{cov}(A_\emptyset) = 1$ (as $A_\emptyset = \mathsf{true}$). Indeed, for $E \neq \emptyset$, the set $\{x' \mid A_E(x')\}$ has $|E|$ fewer degrees of freedom than the support of $\mathcal{D}$, and thus has both measure and coverage equal to zero.

and 2) sparsity results in small I' sets, which further improves convergence.

Minimum Satisfying Assignment This algorithm exploits the duality between MSAs and maximum universal subsets (MUSs): for cost $\mathcal{C}$ and formula $\phi \equiv (B \wedge \mathcal{N}) \rightarrow \hat{y}$, an MUS X is a set of variables with maximum $\mathcal{C}$ such that $\forall X.\phi$, which implies that $C \setminus X$ is an MSA for ϕ [Dillig and others, 2012] and, in turn, an ORE. Thus, the algorithm of [Dillig and others, 2012] focuses on deriving an MUS, and it does so in a recursive branch-and-bound manner, where each branch adds a feature to the candidate MUS. Such an algorithm is exponential in the worst-case, but we mitigated this by selecting a good ordering for feature exploration and performing entailment checks to rule out features that cannot be in the MUS (thus reducing the search tree).

5 Experimental Results

Settings We have trained fully connected (FC) and convolutional neural networks (CNN) models on sentiment analysis datasets that differ in the input length and difficulty of the learning task[6]. We considered 3 well-established benchmarks for sentiment analysis, namely SST [Socher and others, 2013], Twitter [Go and others, 2009] and IMDB [Maas and others, 2011] datasets. From these, we have chosen 40 representative input texts, balancing *positive* and *negative* examples. Embeddings are pre-trained on the same datasets used for classification [Chollet and others, 2015]. Both the HS and MSA algorithms have been implemented in Python and use Marabou [Katz and others, 2019] and Neurify [Wang and others, 2018] to answer robustness queries[7]. In the experiments below, we opted for the kNN-box perturbation space, as we found that the k parameter was easier to interpret and tune than the ϵ parameter for the ϵ-ball space, and improved verification time. Further details on the experimental settings, including a selection of ϵ-ball results, are given in the supplement.

Effect of classifier's accuracy and robustness. We find that our approach generally results in meaningful and compact explanations for NLP. In Figure 2, we show a few OREs extracted for *negative* and *positive* texts, where the returned OREs are both concise and semantically consistent with the predicted sentiment. However, the quality of our OREs depends on that of the underlying classifier. Indeed, enhanced models with better accuracy and/or trained on longer inputs tend to produce higher quality OREs. We show this in Figures 3 and 4, where we observe that enhanced models tend to result in more semantically consistent explanations. For lower-quality models, some OREs include seemingly irrelevant terms (e.g., *"film"*, *"and"*), thus exhibiting shortcomings of the classifier.

Detecting biases As per Prop. 2, we applied exclude constraints to detect biased decisions. In Figure 5, we provide a few example instances exhibiting such a bias, i.e., where *any* robust explanation contains at least one protected feature. These OREs include proper names that shouldn't constitute a sufficient reason for the model's classification. When we try to exclude proper names, no robust explanation exists, indicating that a decision bias exists.

Debugging prediction errors An important use-case for OREs is when a model commits a *misclassification*. Misclassifications in sentiment analysis tasks usually depend on over-sensitivity of the model to polarized terms. In this sense, knowing a minimal, sufficient reason behind the model's prediction can be useful to debug it. As shown in the first example in Figure 6, the model cannot recognize the *double negation* constituted by the terms *not* and *dreadful* as a syntax construct, hence it exploits the negation term *not* to classify the review as *negative*.

Comparison to Anchors We evaluate the robustness of Anchors for FC and CNN models on the SST and Twitter datasets[8]. To compute robustness, we assume a kNN-box perturbation space $\mathcal{B}$ with $k = 15$ for FC and $k = 25$ for CNN models. To extract Anchors, we set $\mathcal{D}$ to the standard perturbation distribution of [Ribeiro and others, 2018], defined by a set of context-wise perturbations generated by a powerful language model. Thus defined $\mathcal{B}$s are small compared to the support of $\mathcal{D}$, and so one would expect high-precision Anchors to be relatively robust w.r.t. said $\mathcal{B}$s. On the contrary, the Anchors extracted for the FC models attain an average precision of 0.996 on SST and 0.975 on Twitter, but only 12.5% of them are robust for the SST case and 7.5% for Twitter. With CNN models, high-quality Anchors are even more brittle: 0% of Anchors are robust on SST reviews and 5.4% on Twitter, despite an average precision of 0.995 and 0.971, respectively.

We remark, however, that Anchors are not designed to provide such robustness guarantees. Our approach becomes useful in this context, because it can *minimally extend* any local explanation to make it robust, by using *include constraints* as explained in Section 3. In Figure 7 we show a few examples of how, starting from non-robust Anchors explanations, our algorithm can find the minimum number of words to make them provably robust.

6 Conclusions

We have introduced optimal robust explanations (OREs) and applied them to enhance interpretability of NLP models. OREs provide concise and sufficient reasons for a particular prediction, as they are guaranteed to be both minimal w.r.t. a given cost function and robust, in that the prediction is invariant for any bounded replacement of the left-out features. We have presented two solution algorithms that build

[6]Experiments were parallelized on a server with two 24-core Intel Xenon 6252 processors and 256GB of RAM, but each instance is single-threaded and can be executed on a low-end laptop.

[7]Marabou is fast at verifying ReLU FC networks, but it becomes memory intensive with CNNs. On the other hand, the symbolic interval analysis of Neurify is more efficient for CNNs. A downside of Neurify is that it is less flexible in the constraint definition (inputs have to be represented as squared bi-dimensional grids, thus posing problems for NLP inputs which are usually specified as 3-d tensors).

[8]Accuracies are 0.89 for FC+SST, 0.82 for FC+Twitter, 0.89 for CNN+SST, and 0.77 for CNN+Twitter.

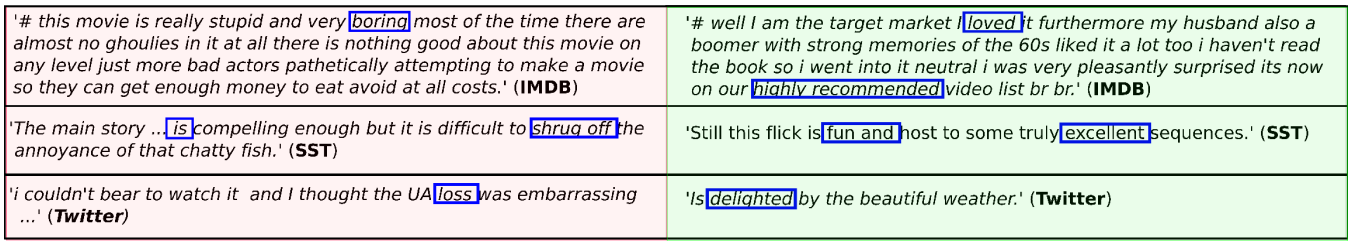

Figure 2: OREs for IMDB, SST and Twitter datasets (all the texts are correctly classified). Models employed are FC with 50 input words each with accuracies respectively 0.89, 0.77 and 0.75. OREs are highlighted in blue. Technique used is kNN boxes with k=15.

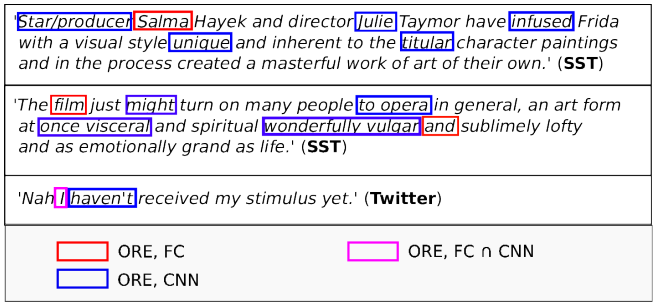

Figure 3: Comparison of OREs for SST and Twitter texts on FC (red) vs CNN (blue) models (common words in magenta). The first two are *positive* reviews, the third is *negative* (all correctly classified). Accuracies of FC and CNN models are, respectively, 0.88 and 0.89 on SST, 0.77 on Twitter. Models have input length of 25 words, OREs are extracted with kNN boxes (k=25).

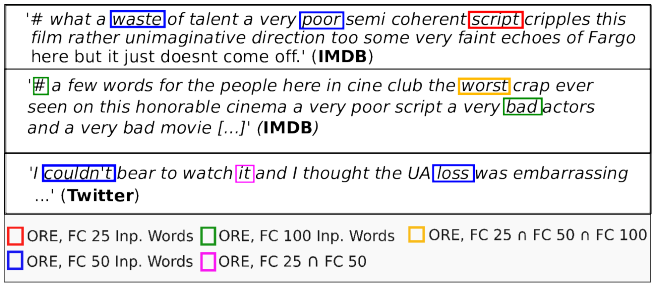

Figure 4: Comparison of OREs on *negative* IMDB and Twitter inputs for FC models. The first and third examples are trained with 25 (red) VS 50 (blue) input words (words in common to both OREs are in magenta). The second example further uses an FC model trained with 100 input words (words in common to all three OREs are in orange). Accuracy is respectively 0.7 and 0.77 and 0.81 for IMDB, and 0.77 for both Twitter models. All the examples are classified correctly. OREs are extracted with kNN boxes (k=25).

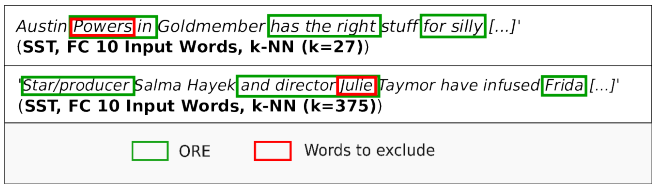

Figure 5: Two examples of decision bias from an FC model with an accuracy of 0.80.

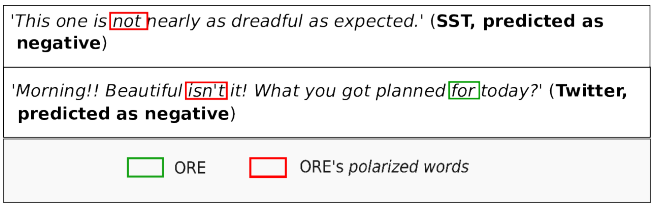

Figure 6: Two examples of over-sensitivity to polarized terms (in red). Other words in the OREs are highlighted in green. Models used are FC with 25 input words (accuracy 0.82 and 0.74). Method used is kNN with k respectively equal to 8 and 10.

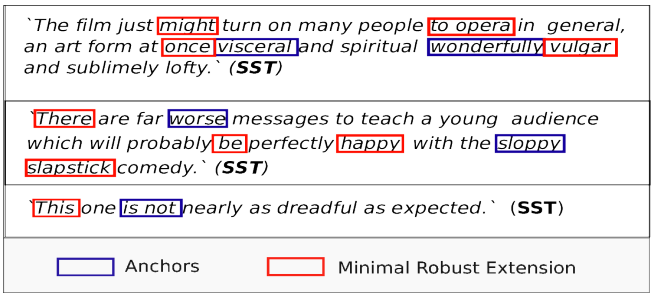

Figure 7: Examples of Anchors explanations (in blue) along with the minimal extension required to make them robust (in red). Examples are classified (without errors) with a 25-input-word CNN (accuracy 0.89). OREs are extracted for kNN boxes and k=25.

on the relation between our OREs, abduction-based explanations and minimum satisfying assignments. We have demonstrated the usefulness of our approach on widely-adopted sentiment analysis tasks, providing explanations for neural network models beyond reach for existing formal explainers. Detecting biased decisions, debugging misclassifications, and repairing non-robust explanations are some of key use cases that our OREs enable. Future research plans include exploring more general classes of perturbations beyond the embedding space.

Acknowledgements This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No. 834115) and the EPSRC Programme Grant on Mobile Autonomy (EP/M019918/1).

References

[Alzantot and others, 2018] M. Alzantot et al. Generating natural language adversarial examples. *arXiv:1804.07998*, 2018.

[Arras and others, 2016] L. Arras et al. Explaining predictions of non-linear classifiers in nlp. *arXiv:1606.07298*, 2016.

[Arras and others, 2017] L. Arras et al. Explaining recurrent neural network predictions in sentiment analysis. *arXiv:1706.07206*, 2017.

[Bach and others, 2015] S. Bach et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *arXiv:1604.00825*, 2015.

[Baroni and others, 2014] M. Baroni et al. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*, pages 238–247, 2014.

[Chakraborty and others, 2017] S. Chakraborty et al. Interpretability of deep learning models: a survey of results. In *SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI*, pages 1–6. IEEE, 2017.

[Chollet and others, 2015] F. Chollet et al. Keras, 2015.

[Croce and others, 2020] F. Croce et al. Sparse-rs: a versatile framework for query-efficient sparse black-box adversarial attacks. *arXiv:2006.12834*, 2020.

[Darwiche and Hirth, 2020] A. Darwiche and A. Hirth. On the reasons behind decisions. *arXiv:2002.09284*, 2020.

[Darwiche, 2020] A. Darwiche. Three modern roles for logic in ai. In *SIGMOD-SIGACT-SIGAI*, pages 229–243, 2020.

[Dillig and others, 2012] I. Dillig et al. Minimum satisfying assignments for SMT. In *CAV*, volume 7358 of *LNCS*, pages 394–409. Springer, 2012.

[Ding and others, 2017] Y. Ding et al. Visualizing and understanding neural machine translation. In *ACL*, pages 1150–1159, 2017.

[Faruqui and others, 2014] M. Faruqui et al. Retrofitting word vectors to semantic lexicons. *arXiv:1411.4166*, 2014.

[Go and others, 2009] A. Go et al. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 2009.

[Huang and others, 2019] P. Huang et al. Achieving verified robustness to symbol substitutions via interval bound propagation. *arXiv:1909.01492*, 2019.

[Ignatiev and others, 2016] A. Ignatiev et al. On finding minimum satisfying assignments. In *CP*, volume 9892 of *LNCS*, pages 287–297. Springer, 2016.

[Ignatiev and others, 2019a] A. Ignatiev et al. Abduction-based explanations for machine learning models. In *AAAI*, pages 1511–1519. AAAI Press, 2019.

[Ignatiev and others, 2019b] A. Ignatiev et al. Model-based diagnosis with multiple observations. In *IJCAI*, pages 1108–1115, 2019.

[Ignatiev and others, 2019c] A. Ignatiev et al. On relating explanations and adversarial examples. In *NeurIPS*, pages 15857–15867, 2019.

[Ignatiev and others, 2019d] A. Ignatiev et al. On validating, repairing and refining heuristic ml explanations. *arXiv:1907.02509*, 2019.

[Jia and others, 2019] R. Jia et al. Certified robustness to adversarial word substitutions. *arXiv:1909.00986*, 2019.

[Katz and others, 2019] G. Katz et al. The Marabou framework for verification and analysis of deep neural networks. In *CAV*, volume 11561 of *LNCS*, pages 443–452. Springer, 2019.

[Ko and others, 2019] C. Ko et al. Popqorn: Quantifying robustness of recurrent neural networks. *arXiv:1905.07387*, 2019.

[La Malfa and others, 2020] E. La Malfa et al. Assessing robustness of text classification through maximal safe radius computation. *"EMNLP 2020 (Findings)"*, 2020.

[Letham and others, 2015] B. Letham et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[Li and others, 2015] J. Li et al. Visualizing and understanding neural models in nlp. *arXiv:1506.01066*, 2015.

[Liu and others, 2018] X. Liu et al. Improving the interpretability of deep neural networks with knowledge distillation. In *ICDMW*, pages 905–912. IEEE, 2018.

[Maas and others, 2011] A. Maas et al. Learning word vectors for sentiment analysis. In *ACL-HLT*, pages 142–150, 2011.

[Mikolov and others, 2013] T. Mikolov et al. Efficient estimation of word representations in vector space. In *ICLR*, 2013.

[Miyato and others, 2016] T. Miyato et al. Adversarial training methods for semi-supervised text classification. *arXiv:1605.07725*, 2016.

[Narodytska and others, 2019] N. Narodytska et al. Assessing heuristic machine learning explanations with model counting. In *SAT*, pages 267–278. Springer, 2019.

[Patel and Bhattacharyya, 2017] K. Patel and P. Bhattacharyya. Towards lower bounds on number of dimensions for word embeddings. In *IJCNLP*, pages 31–36, 2017.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial intelligence*, 32(1):57–95, 1987.

[Ribeiro and others, 2016] M. Ribeiro et al. "why should i trust you?" explaining the predictions of any classifier. In *SIGKDD*, pages 1135–1144, 2016.

[Ribeiro and others, 2018] M. Ribeiro et al. Anchors: High-precision model-agnostic explanations. In *AAAI*, volume 18, pages 1527–1535, 2018.

[Ruan and others, 2018] W. Ruan et al. Reachability analysis of deep neural networks with provable guarantees. *arXiv:1805.02242*, 2018.

[Samek and others, 2017] W. Samek et al. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv:1708.08296*, 2017.

[Shi and others, 2020] W. Shi et al. On tractable representations of binary neural networks. *arXiv:2004.02082*, 2020.

[Singh and others, 2018] C. Singh et al. Hierarchical interpretations for neural network predictions. *arXiv:1806.05337*, 2018.

[Socher and others, 2013] R. Socher et al. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

[Wang and others, 2018] Sh. Wang et al. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, pages 6367–6377, 2018.

[Wang and Rudin, 2015] F. Wang and C. Rudin. Falling rule lists. In *Artificial Intelligence and Statistics*, pages 1013–1022, 2015.

[Zhang and others, 2018] Q. Zhang et al. Unsupervised learning of neural networks to explain neural networks. *arXiv:1805.07468*, 2018.

[Zhang and others, 2019] Q. Zhang et al. Interpreting cnns via decision trees. In *CVPR*, pages 6261–6270, 2019.

7 Appendix

We structure the Appendix in the following way. We first provide proofs of the propositions in Section 3. Second, we give details (through the pseudo-code) of the Algorithms and subroutines that were used to find Optimal Robust Explanations: in particular we describe the *shrink* (used to improve MSA) and the Adversarial Attacks procedures (used to improve HS). We then provide details on the datasets and the architectures that we have used in the Experimental Evaluation, and finally we report many examples of interesting OREs that we were able to extract with our methods, alongside with tables that complete the comparison between MSA and HS as described in the Experimental Evaluation Section.

7.1 Proofs

Proof of Prop. 2 Call $A =$ *"$M(x)$ is biased"* and $B =$ *"(4) is infeasible under $F' \cap E = \emptyset$"*. Let us prove first that $B \rightarrow A$. Note that B can be equivalently expressed as

$$\forall E \subseteq F.(E \cap F' \neq \emptyset \vee \exists x' \in \mathcal{B}_E(t).M(x) \neq M(x'))$$

If the above holds for all E then it holds also for $E = F \setminus F'$, and so it must be that $\exists x' \in \mathcal{B}_{F \setminus F'}(t).M(x) \neq M(x')$ because the first disjunct is clearly false for $E = F \setminus F'$.

We now prove $A \rightarrow B$ by showing that $\neg B \rightarrow \neg A$. Note that $\neg B$ can be expressed as

$$\exists E \subseteq F.(E \cap F' = \emptyset \wedge \forall x' \in \mathcal{B}_E(t).M(x) = M(x')), \quad (6)$$

and $\neg A$ can be expressed as

$$\forall x' \in \mathcal{B}_{F \setminus F'}(t).M(x) = M(x'). \quad (7)$$

To see that (6) implies (7), note that any E that satisfies (6) must be such that $E \cap F' = \emptyset$, which implies that $E \subseteq F \setminus F'$, which in turn implies that $\mathcal{B}_{F \setminus F'}(t) \subseteq \mathcal{B}_E(t)$. By (6), the prediction is invariant for any x' in $\mathcal{B}_E(t)$, and so is for any x' in $\mathcal{B}_{F \setminus F'}(t)$.

Proof of Prop. 3 A robust explanation $E \subseteq F$ guarantees prediction invariance for any $x' \in \mathcal{B}_E(t)$, i.e., for any x' (in the support of $\mathcal{D}$) to which anchor A_E applies.

Proof of Prop. 4 For discrete $\mathcal{D}$ with pmf $f_{\mathcal{D}}$, we can express $\mathsf{cov}(A_E)$ as

$$\mathsf{cov}(A_E) = \sum_{x' \in supp(\mathcal{D})} f_{\mathcal{D}}(x') \cdot \mathbf{1}_{A_E(x')} = \sum_{x' \in supp(\mathcal{D})} f_{\mathcal{D}}(x') \cdot \prod_{w \in E} \mathbf{1}_{x'_w = \mathcal{E}(w)}$$

To see that, for $E' \supseteq E$, $\mathsf{cov}(A_{E'}) \leq \mathsf{cov}(A_E)$, observe that $\mathsf{cov}(A_{E'})$ can be expressed as

$$\mathsf{cov}(A_{E'}) = \sum_{x' \in supp(\mathcal{D})} f_{\mathcal{D}}(x') \cdot \prod_{w \in E'} \mathbf{1}_{x'_w = \mathcal{E}(w)} = \sum_{x' \in supp(\mathcal{D})} f_{\mathcal{D}}(x') \cdot \prod_{w \in E} \mathbf{1}_{x'_w = \mathcal{E}(w)} \cdot \prod_{w \in E \setminus E'} \mathbf{1}_{x'_w = \mathcal{E}(w)}$$

and that for any x', $\prod_{w \in E \setminus E'} \mathbf{1}_{x'_w = \mathcal{E}(w)} \leq 1$.

Proof of Prop. 1 With abuse of notation, in the following we use C^* to denote both an ORE and its logical encoding.

1. if C^* is an ORE, then $\phi \equiv (B \wedge \mathcal{N}) \rightarrow \hat{y}$ is true for *any* assignment x' of the features not in C^*. In particular, ϕ is trivially satisfied for any x' outside the perturbation space B, and, by Definition 1, is satisfied for any x' within the perturbation space.
2. As also explained in [Dillig and others, 2012], finding an optimal C^* such that $C^* \models \phi$ is equivalent to finding an MSA C^* for ϕ. We should note that C^* is a special case of an MSA, because the possible assignments for the variables in C^* are restricted to the subsets of the cube C.
3. C^* is said a prime implicant of ϕ if $C^* \models \phi$ and there are no proper subsets $C' \subset C^*$ such that $C' \models \phi$. This holds regardless of the choice of the cost $\mathcal{C}$, as long as it is additive and assigns a positive cost to each feature as per Definition 2. Indeed, for such a cost function, any proper subset $C' \subset C^*$ would have cost strictly below that of C^*, meaning that $C' \not\models \phi$ (i.e., is not robust) because otherwise, C' (and not C^*) would have been (one of) the robust explanations with minimal cost.

7.2 Optimal Cost Algorithms and Sub-Routines

In this Section we provide a full description and the pseudo-code of the algorithms that for reason of space we were not able to insert in the main paper. We report a line-by-line description of the HS procedure (Algorithm 1): we further describe how the adversarial attacks procedure is used to generate candidates that help the HS approach converge on hard instances, as reported in Section 4. We then describe the algorithm to compute Smallest Cost Explanations (Algorithm 4). In Algorithm 5, we finally detail the *shrink* procedure as sketched in Section 3.

Minimal Hitting-Sets and Explanations One way to compute optimal explanations against a cost function C, is through the hitting set paradigm [Ignatiev and others, 2019a], that exploits the relationship between diagnoses and conflicts [Reiter, 1987]: the idea is to collect perturbations and to calculate on their indices a minimum hitting set (MHS) i.e., a minimum-cost explanation whose features are in common with all the others. We extend this framework to find a word-level explanation for non-trivial NLP models. At each iteration of Algorithm 1, a minimum hitting set E is extracted (line 3) from the (initially empty, line 1) set Γ. If function *Entails* evaluates to *False* (i.e., the neural network $\mathcal{N}$ is provably safe against perturbations on the set of features identified by $F \setminus F$) the procedure terminates and E is returned as an ORE. Otherwise, (at least) one feasible attack is computed on $F \setminus E$ and added to Γ (lines 7-8): the routine then re-starts. Differently from [Ignatiev and others, 2019a], as we have experienced that many OREs whose a large perturbation space - i.e. when ϵ or k are large - do not terminate in a reasonable amount of time, we have extended the *vanilla* hitting set approach by introducing *SparseAttacks* function (line 7). At each iteration *SparseAttacks* introduces in the hitting set Γ a large number of sparse adversarial attacks on the set of features $F \setminus E$: it is in fact known [Ignatiev and others, 2016] that

attacks that use as few features as possible help convergence on instances that are hard (intuitively, a small set is harder to "hit" hence contributes substantially to the optimal solution compared to a longer one) *SparseAttacks* procedure is based on random search and it is inspired by recent works in image recognition and malaware detection [Croce and others, 2020]: pseudo-code is reported in 2, while a detailed description follows in the next paragraph.

Sparse Adversarial Attacks In Algorithm 2 we present a method to generate sparse adversarial attacks against features (i.e., words) of a generic input text. $GeneratePerturbations(k, n, Q)$ (line 2) returns a random population of n perturbations that succeed at changing $\mathcal{N}$'s classification: for each successful attack p, a subset of k out of d features has been perturbed through a Fast Gradient Sign attack [9] (FGSM), while it is ensured that the point lies inside a convex region Q which in our case will be the ϵ hyper-cube around the embedded text. If no perturbation is found in this way (i.e., population size of the atacks is zero, as in line 3), budget is decreased (line 4) and another trial of $GeneratePerturbations(k, n, Q)$ is performed (e.g., with few features as targets and a different random seed to guide the attacks). Function $AccuracyDrop(\mathcal{N}, P)$ returns the best perturbation a where k is increasingly minimised (line 7). Algorithm terminates when either no attacks are possible (all the combinations of features have been explored) or after fixed number of iterations has been performed (line 1).

Algorithm 1: ORE computation via implicit hitting sets and sparse attacks

Data: a network $\mathcal{N}$, the input text t, the initial set of features F, a network prediction $\hat{y}$, a cost function $\mathcal{C}$ against which the explanation is minimised
Result: an optimal ORE E

1 $\Gamma = \emptyset$
2 **while** *true* **do**
3 $\quad E = MinimumHS(\Gamma, \mathcal{C})$
4 $\quad$ **if** $Entails(E, (\mathcal{N} \wedge \mathcal{B}_{F \setminus E}(t)) \rightarrow \hat{y})$ **then**
5 $\quad\quad$ **return** E
6 $\quad$ **else**
7 $\quad\quad A = SparseAttacks(E, \mathcal{N})$
8 $\quad\quad \Gamma = \Gamma \cup \{A\}$

Minimum Satisfying Assignment Explanations This approach, based on the method presented in [Dillig and others, 2012], finds an explanation in the form of an MSA, for which in turn a maximum universal subset (MUS) is required. For a given cost function $\mathcal{C}$ and text t, an MUS is a universal subset t' of words that maximises $\mathcal{C}(t')$. An MSA of the network M w.r.t the text is precisely a satisfying assignment of the formula $\forall_{w \in t'}.M \rightarrow \hat{y}$ for some MUS t'. In other words, an MSA is $t \setminus t'$. The inputs to the MSA algorithm are: $\mathcal{N}$ which

[9] https://www.tensorflow.org/tutorials/generative/adversarial_fgsm

Algorithm 2: Computing a perturbation that is successful and minimises the number of features that are perturbed.

Data: $\mathcal{N}$ - neural network model, F - input text from feature space; $k \in \mathbb{N}^{+}_{\setminus\{0\}}$ - number of perturbations initially tested; $Q \subseteq F$ - (sub)set of features where perturbations are found; $n \in \mathbb{N}^{+}_{\setminus\{0\}}$ - number of elements generated at each iteration of the algorithm; budget - number of iterations allowed before stopping.

1 **while** $k > 0 \ \wedge \ budget > 0$ **do**
2 $\quad P \leftarrow GeneratePerturbations(k, n, Q)$
3 $\quad$ **if** *length(P)* $== 0$ **then**
4 $\quad\quad budget \leftarrow budget - 1$
5 $\quad\quad$ continue
6 $\quad$ **end**
7 $\quad a \leftarrow \arg\max_{p \in P} AccuracyDrop(M, P)$
8 $\quad k \leftarrow k - 1, budget \leftarrow budget - 1$
9 **end**
10 return a

represents the network M in constraint form; text t; cost function $\mathcal{C}$ and prediction $\hat{y}$ for the input t. The algorithm first uses the reversed sort function for the text t to optimize the search tree. The text is sorted by the cost of each word. then uses the recursive MUS algorithm to compute an MUS t'. Finally, the optimal explanation ($t \setminus t'$) is returned.

The inputs of the mus algorithm are: a set of candidate words cW that an MUS should be calculated for (equal to t in the first recursive call), a set of bounded words bW that may be part of an MUS, where $\forall_{w \in bW}$, w may be limited by ϵ-ball or k-NN box clousure, a lower bound L, the network $\mathcal{N}$, a cost function $\mathcal{C}$, and a network prediction $\hat{y}$. It returns a maximum-cost universal set for the network $\mathcal{N}$ with respect to t, which is a subset of cW with a cost greater than L, or the empty set when no such subset exists. The lower bound allows us to cut off the search when the current best result cannot be improved. During each recursive call, if the lower bound cannot be improved, the empty set is returned (line 1). Otherwise, a word w is chosen from the set of candidate words cW and it is determined whether the cost of the universal subset containing word w is higher than the cost of the universal subset without it (lines 5-12). Before definitively adding word w to bW, we test whether the result is still satisfiable with $Entails$ (line 5) i.e. still an explanation. The $shrink$ method helps to reduce the set of candidate words by iterating through current candidates and checking using $Entails$ whether they are necessary. This speeds-up the algorithm (as there are fewer overall calls to $Entails$). The recursive call at line 6 computes the maximum universal subset of $\forall_{w \in bW} \mathcal{N} \rightarrow \hat{y}$, with adjusted cW and L as necessary. Finally within this if block, we compute the cost of the universal subset involving word w, and if it is higher than the previous bound L, we set the new lower bound to cost (lines 7-11). Lines 11-12 considers the cost of the universal subset *not* containing word w, in case it has higher cost, and if so,

Algorithm 3: MUS computation, $mus(bW, \mathcal{N}, cW, t, \mathcal{C}, L, \hat{y})$

Data: a list of bounded words bW, a network $\mathcal{N}$, a set of candidate words cW, the input text t, a cost function $\mathcal{C}$ against which the ORE is minimised, a lower bound for MUS L, a prediction $\hat{y}$ for the input
Result: a Maximum Universal Subset with respect to input text t

```
if cW = ∅ or C(cW) ⩽ L then return ∅
best = ∅
choose w ∈ cW
bW = bW ∪ {w}, constW = cW \ {w}
if Entails(constW, (N ∧ B_{F\E}(constW)) → ŷ) then
    Y = mus(bW, N, shrink(N, bW, cW \ {w}), t, C, L − C(w), ŷ)
    cost = C(Y) + C(w)
    if cost > L then
        best = Y ∪ {w}
        L = cost
Y = mus(bW \ {w}, N, cW \ {w}, t, C, L, ŷ)
if C(Y) > L then best = Y
return best
```

updates $best$. Once one optimal explanation has been found, it is possible to compute *all combinations* of the input that match that cost, and then use $Entails$ on each to keep only those that are also explanations.

Comparing MHS and MSA The MSA-based approach uses MUS algorithm to find maximum universal subset and then finds a MSA for that MUS. MUS is a recursive branch-and-bound algorithm [Dillig and others, 2012] that explores a binary tree structure. The tree consists of all the word appearing in the input cube. The MUS algorithm possibly explores an exponential number of universal subsets, however, the recursion can be cut by using right words ordering (i.e. words for which robustness query will answer false, consider words with the highest cost first) or with shrink method. MUS starts to work with a full set of candidate words, whereas the HS approach starts with an empty set of fixed words and tries to find an attack for a full set of bounded words. In each iteration step, the HS approach increases the set of fixed words and tries to find an attack. It is because a subset $t' \subseteq t$ is an MSA for a classifier M with respect to input text t iff t' is a minimal hitting set of minimum falsifying set (see [Ignatiev and others, 2016] for details). To speed up the MSA algorithm, we use $shrink$ procedure which reduces the set of candidate words, and for non-uniform cost function, words ordering (words with the highest cost are considered as the first candidates), while HS-based approach uses $SparseAttacks$ routine to increase the hitting set faster.

Excluding words from MSA To exclude specific words from a smallest explanation we add one extra argument to the MSA algorithm input: the bW which represents bounded words. In this case the set $cW = t \setminus bW$. From now on the procedure is the standard one.

Algorithm 4: Computing smallest cost explanation

Data: a network $\mathcal{N}$, an input text t, a cost function $\mathcal{C}$ for the input C, a prediction $\hat{y}$,
Result: A smallest cost explanation for network $\mathcal{N}$ w.r.t. input text t

```
bW = ∅, cW = C, sce = ∅
textSortedByCost = sort(t)
maxus = mus(bW, N, cW, textSortedByCost, C, 0, ŷ)
foreach c ∈ t do
    if c ∉ maxus then
        sce = sce ∪ c
    end
end
return sce
```

Algorithm 5: $shrink$ algorithm $shrink(bW, \mathcal{N}, cW, C, \mathcal{C}, L, \hat{y})$

Data: a list of bounded words bW, a network $\mathcal{N}$, a set of candidate words cW, a text t, a cost function $\mathcal{C}$, a lower bound L, a prediction $\hat{y}$ for the input
Result: A set of the essential candidate words eW

```
eW = cW
foreach word ∈ cW do
    eW = eW \ {word}
    bW = bW ∪ {word}
    constW = C \ bw
    if Entails(constW, (N ∧ B_{F\E}(cW)) → ŷ) then
        eW = eW ∪ {word}
    end
    bW = bW \ {word}
end
return eW
```

7.3 Details on the Experimental Results

Datasets and Test Bed

As mentioned in the Experimental Evaluation Section, we have tested MSA and HS approaches for finding optimal cost explanations respectively on the SST, Twitter and IMDB datasets. For each task, we have selected a sample of 40 input texts that maintain classes balanced (i.e., half of the examples are *negative*, half are *positive*). Moreover, we inserted inputs whose polarity was exacerbated (either very *negative* or very *positive*) as well as more challenging examples that machines usually misclassifies, like *double negations* or mixed sentiments etc. Further details in Table 1.

Models Setup

We performed our experiments on FC and CNNs with up to 6 layers and $20K$ parameters. FC are constituted by a stack of *Dense* layers, while CNNs additionally employ *Convolutional* and *MaxPool* layers: for both CNNs and FC the decision is taken through a *softmax* layer, with *Dropout* that is addedd after each layer to improve generalization during

'# I've seen Foxy Brown, Coffy Friday Foster Bucktown, and Black Mama White Mama of these this is Pam Griers worst movie poor acting bad script boring action scenes theres just nothing there avoid this and rent Friday Foster Coffy or Foxy Brown instead' (**IMDB, predicted as *negative***)
'# I gave this a 2 and it only avoided a 1 because of the occasional unintentional laugh the film is excruciatingly. Boring and incredibly cheap its even worse if you know anything at all about the Fantastic Four.', (***IMDB, predicted as negative***)
'# a few words for the people here in cine club the worst crap ever seen on this honorable cinema a very poor script a very bad actors and a very bad movie dont waste your time looking this movie see the very good or any movie have been good commented by me say no more' (**IMDB, predicted as *negative***)

Figure 8: Examples of Optimal Robust Explanations - highlighted in blue -. OREs were extracted using kNN boxes with 25 neighbors per-word: fixing words in an ORE guarantees the model to be locally robust. The examples come from the IMDB dataset, model employed is a FC network with 100 input words (accuracy 0.81).

the training phase. As regards the embeddings that the models equip, we experienced that the best trade-off between the accuracy of the network and the formal guarantees that we need to provide is reached with low-dimensional embeddings, thus we employed optimized vectors of dimension 5 for each word in the embedding space: this is in line with the experimental evaluations conducted in [Patel and Bhattacharyya, 2017], where for low-order tasks such as sentiment analysis, compact embedding vectors allow to obtain good performances, as shown in Table 1. We note that techniques such as retro-fitting [Faruqui and others, 2014] could allow using more complex representations and might help with high-order tasks such as multi-class classification, where the quality of the embedding plays a crucial role in terms of accuracy. We will consider this as a future extension of the work. We report in Table 1 an overview of the models we used.

7.4 Additional Results

In this Section we provide a few interesting results that we couldn't add to the main paper.

Additional kNN Results

As discussed in Section 5, we have found a few instances where distilling an ORE from an ϵ-bounded input was computationally infeasible, thus motivating us to develop and use the kNN-boxes technique for the majority of the results in this paper. In Figure 10 we compare how OREs grow for increasing values of ϵ and k (i.e., the control parameters of respectively ϵ-boxes and kNN-boxes). Finally, in Figure 8 we report a few examples of IMDB reviews that we could solve for an FC with 100 input words: those examples show OREs for the largest model - in terms of both input size and parameters - that we could solve by means of HS or MSA, eventually improved with the *Adversarial Attacks* routine.

ϵ-ball Results

With a perturbation method defined as an ϵ-ball around each input vector (see section 3), Table 2 shows a comparison of ORE length and execution time for both the MSA and HS methods.

Figure 9 shows how using adversarial attacks speeds up convergence.

Below is an example of calculating all possible OREs for a given input and ϵ, and an example of decision bias.

	INPUT INSTANCE	EXECUTION TIME [s] (HS *Vanilla*, HS + Adversarial Attacks, MSA)
ε = 0.05	'insanely hilarious!'	**87.66**, **8.67**, **47.56**
ε = 0.05	'this one is not nearly as dreadful as expected'	**114.99**, **10.49**, **0.44**
ε = 0.05	'this one is baaaaad movie!'	**Timeout**, **79.2**, **0.79**
ε = 0.1	'so your entire day was spent doing chores ay??!!' [...]	**Timeout**, **1520.80**, **0.44**
ε = 0.05	'I just seen ur tweetz plz write bak' [...]	**Timeout**, **159.11**, **930.83**

Figure 9: Examples of explanations that were enabled by the adversarial attacks routine. Timeout was set to 2 hours.

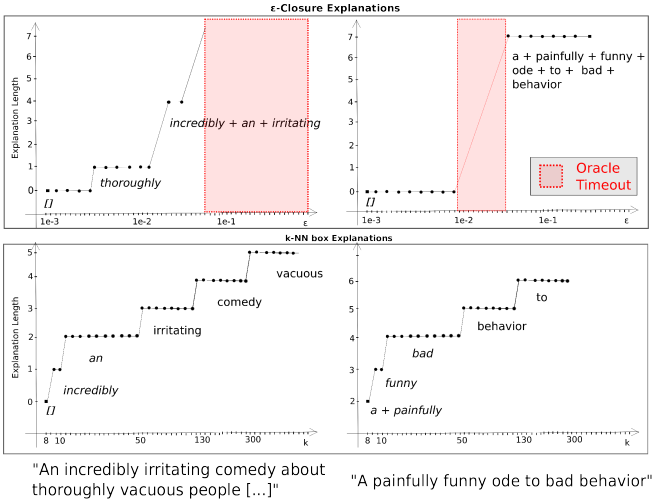

Figure 10: How an explanation grows when either ϵ (top) or k (bottom) is increased. Model considered is a fully connected with 50 input words on SST dataset (0.89 accuracy). On the left a *negative* review that is correctly classified, on the right a *positive* review that is misclassified (i.e., the model's prediction is *negative*). For specific ranges of ϵ the Oracle cannot extract an explanation (timeout, highlighted in red).

Table 1.1: Training

	TWITTER	SST	IMDB
Inputs (Train, Test)	$1.55M, 50K$	$117.22K, 1.82K$	$25K, 25K$
Output Classes	2	2	2
Input Length (max, max. used)	88, 50	52, 50	2315, 100
Neural Network Models	FC, CNN	FC, CNN	FC, CNN
Neural Network Layers (min,max)	3,6	3,6	3,6
Accuracy on Test Set (min, max)	0.77, 0.81	0.82, 0.89	0.69, 0.81
Number of Networks Parameters (min,max)	$3K, 18K$	$1.3K, 10K$	$5K, 17K$

Table 1.2: Explanations

	TWITTER	SST	IMDB
Sample Size	40	40	40
Review Length (min-max)	10, 50	10, 50	25, 100

Table 1: Datasets used for training/testing and extracting explanations. We report various metrics concerning the networks and the training phase (included accuracy on Test set), while in Table 1.2 we report the number of texts for which we have extracted explanations along with the number of words considered when calculating OREs: samples were chosen to reflect the variety of the original datasets, i.e., a mix of long/short inputs equally divided into positive and negative instances.

ϵ	Explanation Length	MSA Execution Time	HS Execution Time
0.01	5 ± 5	8.08 ± 7.9	63.70 ± 63.69
0.05	5.5 ± 4.5	176.22 ± 175.92	339.96 ± 334.66
0.1	7.5 ± 2.5	2539.75 ± 2539.14	3563.4 ± 3535.84

Table 2: Comparison between MSA and HS in terms of execution time for different values of ϵ, and the corresponding explanation length.

Example 1 Calculating *all* of the smallest explanations for an input ($\epsilon = 0.05$, FC network, 10 input words, 5 dimensional embedding, SST dataset):

```
Input: ['strange', 'funny', 'twisted', 'brilliant',
     'and', 'macabre', '<PAD>', '<PAD>', '<PAD>',
    '<PAD>']

 Explanations (5 smallest, len=6.0): ['strange', '
     funny', 'twisted', 'brilliant', '<PAD>', '<PAD
     >'] ['strange', 'funny', 'twisted', '<PAD>',
     '<PAD>', '<PAD>'] ['strange', 'twisted', '
     brilliant', '<PAD>', '<PAD>', '<PAD>'] ['
     strange', 'twisted', 'and', '<PAD>', '<PAD>',
     '<PAD>'] ['strange', 'twisted', 'macabre', '<
     PAD>', '<PAD>', '<PAD>']
```

Example 2 Decision bias, as *Derrida* cannot be excluded ($\epsilon = 0.05$, FC network, 10 input words, 5 dimensional embedding, SST dataset):

```
Input: ['Whether', 'or', 'not', 'you', 'are', '
    enlightened', 'by', 'any', 'of', 'Derrida']

Exclude: ['Derrida']

Explanation: ['Whether', 'or', 'are', 'enlightened
    ', 'by', 'any', 'Derrida']
```