Sensitivity as a Complexity Measure for Sequence Classification Tasks

Michael Hahn
Stanford University,
United States
mhahn2@stanford.edu

Dan Jurafsky
Stanford University,
United States
jurafsky@stanford.edu

Richard Futrell
University of California, Irvine,
United States
rfutrell@uci.edu

Abstract

We introduce a theoretical framework for understanding and predicting the complexity of sequence classification tasks, using a novel extension of the theory of Boolean function sensitivity. The sensitivity of a function, given a distribution over input sequences, quantifies the number of disjoint subsets of the input sequence that can each be individually changed to change the output. We argue that standard sequence classification methods are biased towards learning low-sensitivity functions, so that tasks requiring high sensitivity are more difficult. To that end, we show analytically that simple lexical classifiers can only express functions of bounded sensitivity, and we show empirically that low-sensitivity functions are easier to learn for LSTMs. We then estimate sensitivity on 15 NLP tasks, finding that sensitivity is higher on challenging tasks collected in GLUE than on simple text classification tasks, and that sensitivity predicts the performance both of simple lexical classifiers and of vanilla BiLSTMs without pretrained contextualized embeddings. Within a task, sensitivity predicts which inputs are hard for such simple models. Our results suggest that the success of massively pretrained contextual representations stems in part because they provide representations from which information can be extracted by low-sensitivity decoders.

1 Introduction

What makes some tasks harder and others easier for modern machine learning methods?[1] In NLP, simple models based on lexical classifiers provide good performance on some tasks, while strong performance on other tasks has been attained only recently with massive pretrained models. However, there is no unified theoretical framework for understanding these difficulty differences between tasks, or what models might be more or less effective.

[1]Code: https://github.com/m-hahn/sensitivity.

Existing complexity metrics provide limited practical insight. The Chomsky Hierarchy (Chomsky, 1956) is a prominent classification of formal languages by complexity, but it describes asymptotic worst-case complexity and does not provide a measure of how hard it is to achieve high accuracy on realistic task distributions. Kolmogorov complexity (Li and Vitányi, 1993) is uncomputable and becomes well-defined only in the asymptotic limit. Psycholinguistic complexity metrics such as surprisal (Hale, 2001) and dependency length (Gibson, 1998) only capture formal features of the input, without regard to the task.

We propose **sensitivity** as a theory of complexity for sequence classification tasks, that is, any task involving learning a function from sequences to labels. The sensitivity of a function, given a distribution over input sequences, quantifies the number of disjoint subsets of the input sequence that can each be individually changed in such a way as to change the output. Intuitively, high-sensitivity functions are complex because a single change in the input, in many different places, can completely change the output; low-sensitivity functions are simpler because the output is predictable from redundant information in many subsets of the input. We will argue that sensitivity predicts what tasks are easy or hard for modern machine learning methods to learn.

Our notion of sensitivity is grounded in a well-studied theory for Boolean functions (O'Donnell, 2014), which we generalize to natural language. Unlike measures like Kolmogorov complexity, sensitivity can be estimated on real datasets and single inputs without asymptotic approximations, only requiring a generalized language model such as XLNet (Yang et al., 2019) and a strong model of the task.

In this paper, we argue that sensitivity captures informal notions of complexity both at the level of architectures and on the level of tasks. First, we show that sensitivity quantifies architectural limitations and inductive biases of various machine

Transactions of the Association for Computational Linguistics, vol. 9, pp. 891–908, 2021. https://doi.org/10.1162/tacl_a_00403
Action Editor: Annie Louis. Submission batch: 2/2021; Revision batch: 4/2021; Published 8/2021.
© 2021 Association for Computational Linguistics. Distributed under a CC-BY 4.0 license.

learning architectures used in NLP, including both lexical classifiers and vanilla LSTMs without pretrained contextualized embeddings (Section 3). Second, in a survey of 15 major NLP tasks, we find that sensitivity quantitatively predicts how difficult a task is for simple lexical classifiers and neural models, both across tasks and across different inputs for a single task (Section 4). The validity of our methods for quantifying sensitivity is verified using human experiments in Section 5. Section 6 discusses the relationship of sensitivity to previous theories of complexity and brittleness in neural networks, and implications for NLP practice. Section 7 concludes.

2 Sensitivity

2.1 Analysis of Boolean Functions

We build on notions of sensitivity developed for Boolean functions (Kahn et al., 1988; Hatami et al., 2010; O'Donnell, 2014). Analysis of Boolean functions is a powerful and rigorous theory with wide-ranging applications in theoretical computer science (O'Donnell, 2014). We first introduce the relevant notions, and then explain how these concepts can be generalized to the setting of fully general sequence classification. The **sensitivity** of a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ at a bitstring $x \in \{-1,1\}^n$ is defined as:

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}, \tag{1}$$

where $x^{\oplus i}$ is the result of flipping the i-th bit of x. This describes how many bits of x can be flipped individually to change f, or equivalently, how many Hamming neighbors of x have the opposite value of f.

The highest possible sensitivity is attained by the PARITY function $f_{\text{Parity}}(x) := \prod_{i=1}^{n} x_i$. Given a string of ''1''s and ''-1''s, this function counts whether the number of negative inputs is even (output $+1$) or odd (output -1). For instance, $f_{\text{Parity}}(1,1,1) = f_{\text{Parity}}(1,-1,-1) = 1$ and $f_{\text{Parity}}(1,-1,1) = f_{\text{Parity}}(-1,1,1) = -1$. The function f_{Parity} has the property that flipping any individual bit flips the output. For instance, given the string ''1 1 1'', changing any of the three input symbols to ''-1'' flips the parity of the string from $+1$ to -1. Therefore, for every bitstring $x \in \{-1,1\}^n$, we have $s(f_{Parity}, x) = n$. It is impossible to approximate f_{Parity} beyond chance level with linear functions (Minsky and Papert, 1969), or with linear combinations of functions that contain nonlinear interactions between less than n input bits (O'Donnell, 2014). In this sense, the function f_{Parity} is maximally nonlinear. On the other hand, low-sensitivity functions can be approximated with linear functions or linear combinations of functions that each only combine a few input bits (O'Donnell, 2014, Thm. 2.38). Sensitivity also has close connections with other complexity measures such as decision tree depth (Nisan, 1991) and the degree of a Boolean function when written as a polynomial.

2.2 Application to sequence classification

We argue that this theory can be brought to bear to quantify the complexity of sequence classification tasks. In this setting, sensitivity measures the nonlinearity of the decision boundary. Low sensitivity tasks are those where simple methods based on linear combinations of local features are most successful. For instance, low sensitivity tasks can be solved by bag-of-words classifiers and linear classifiers based on n-gram features, which have bounded similarity (as we will make precise in Proposition 1 below). On the other hand, high sensitivity tasks require more sophisticated methods. We expect that tasks that have proven empirically difficult in the literature, such as those requiring reasoning, correspond to those with high sensitivity, which means that changing different substrings in an input can easily flip the label (e.g., ENTAILMENT ⇒ NONENTAILMENT).

Testing these ideas requires generalizing sensitivity to functions more akin to those relevant in NLP along several aspects. One aspect can be dealt with without major changes: NLP tasks are defined on alphabets Σ with more than two elements, such as the words of a language. The theory can be accommodated to such alphabets, leading to a generalized definition of sensitivity applicable when the symbols X_i are distributed independently and uniformly (rephrased based on O'Donnell, 2014, Def. 8.22):

$$s(f,x) := \sum_{i=1}^{n} \text{Var}\left(f(X) | \forall j \neq i : X_j = x_j\right), \tag{2}$$

where the variance measures how much f varies across strings $X \in \Sigma^n$ that agree with x on all except possibly the i-th input. Definition (2)

Sentence 1:
a gorgeous , witty , seductive movie . (+1)
seductive (+1), brilliant (+1), cute (+1), sexy (2×, +1), shocking (2×, +1), stylish (+1), charming (+1)
Sentence 2:
a painfully funny ode to bad behavior (+1)
bleak (−1), small (−1), accurate (2×, −0.2), sexy (0.13), true (0.5), beautiful (0.9) funny (2×, +1), honest (+1)

Figure 1: Subset sensitivity (3) for sentiment analysis, for two inputs from the SST-2 dev set. For each inputs, we select a one-word subsequence (marked in blue, corresponding to sets $\{2\}$ for Sentence 1, and $\{3\}$ for Sentence 2), and show 10 possible substitutions sampled using XLNet (see Section 4; "2×" indicates samples appearing twice). We show sentiment prediction (between -1.0 for negative and $+1.0$ for positive sentiment), obtained using RoBERTa (see Section 4), both for the original sentence and each version arising from substituting any of the other adjectives. In Sentence 1, due to the presence of positive adjectives in the context, the distribution is concentrated on positive adjectives; $f(x') = +1$ for each sampled $x' \in x^{\oplus P}$. Therefore, subset sensitivity $s(f, x, P)$ is estimated as 0.0. In Sentence 2, both positive and negative adjectives are plausible substitutions, and $s(f, x, P) = 0.58$.

reduces to (1) if $\Sigma = \{-1, 1\}$ and $f : \{-1, 1\}^n \to \{-1, 1\}$.

More challenging is the fact that symbol sequences in language are not distributed uniformly. For example, in movie review sentiment classification, most inputs will sound like movie reviews (rather than tweets or Wikipedia articles), and almost all will respect the grammatical and statistical properties of the underlying language. When defining a generalization of $s(f, x)$ to natural language, we want to focus on those strings x and their Hamming neighbors x' that are typical instances of the problem. We next describe an adaptation of Equations (1) and (2) taking this into account.

2.3 Formal Definitions

In order to adapt the idea of sensitivity to the setting of NLP tasks, we introduce a generalized notion called block sensitivity. Block sensitivity is the maximum sensitivity over all possible partitions of the input bits. Block sensitivity has been studied for Boolean functions as an upper bound on (1) (Nisan, 1991; Bernasconi, 1996; Hatami et al., 2010); we construct a probabilistic version of this notion as a sensitivity measure appropriate to more sequence classification tasks.

Consider a set Σ (e.g., the words of a language), with an arbitrary distribution Π over the set Σ^* of finite sequences of symbols from Σ. We formalize classification tasks as functions $f : \Sigma^* \to [-1, 1]$.[2] Such functions could be binary classifiers f mapping to $\{-1, 1\}$, or they could output a continuous score. We take the output space to be $[-1, 1]$ instead of $[0, 1]$ to make our definitions consistent with those from the analysis of Boolean functions.

The **subset sensitivity** of the function $f : \Sigma^* \to \mathbb{R}$ on the point $x \in \Sigma^n$ and the set $P \subset \{1, \ldots, n\}$ is

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right), \quad (3)$$

where $x^{\oplus P}$ denotes the set of all strings x' that agree with x on all indices outside of P:

$$x^{\oplus P} := \{x' \in \Sigma^n : x'_j = x_j \text{ for all } j \in \{1, \ldots, n\} - P\}, \quad (4)$$

and the variance is computed with respect to Π. If P is a singleton $\{i\}$, we recover the term inside the sum in (2): $s(f, x) = \sum_{i=1}^{n} s(f, x, \{i\})$.

We illustrate this definition in Figure 1 with examples from the Stanford Sentiment Treebank (Socher et al., 2013). Here, the function f maps movie reviews to the probability that the review is positive, scaled to $[-1, 1]$. For each sentence, we select a singleton subset P and show 10 samples from Π, the distribution over possible substitutions. In Sentence 1, due to the positive adjectives in the context, the distribution is concentrated on positive adjectives, and so the sensitivity $s(f, x, P) \approx 0$. In Sentence 2, both positive and negative adjectives are plausible substitutions, and $s(f, x, P) \approx 0.6$.

This example shows how (3) differs from the vanilla definition (1) by accounting for the statistical dependencies between words in natural language: It takes into account that the choice of possible completions for a set P is often constrained by the context given by x. Inputs violating

[2] For multi-class problems, we take a family of functions f corresponding to the classes, see Section 4.

these statistical dependencies (e.g., 'a *boring*, witty, seductive movie' for Figure 1) are unlikely to occur in naturalistic input, and the behavior of f on such unlikely inputs may not impact the difficulty of representing f with high average fidelity. This motivates considering the variance of f over neighboring strings, instead of, say, the entire range of f over all possible neighboring strings.

Based on subset sensitivity, we introduce the **block sensitivity** at x as an analogue to (1):

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i), \quad (5)$$

where the maximization ranges over all partitionings of $\{1,\ldots,n\}$ into disjoint subsets $P_1 \dot\cup \ldots \dot\cup P_k$ ($\dot\cup$ denoting disjoint union). We recover the quantity $s(f,x)$ in (1) and (2) by restricting subsets P_i to the singletons $\{i\}$; thus, we have

$$bs(f,x) \geq s(f,x). \quad (6)$$

Intuitively, $bs(f,x)$ measures the following: Given an input x, how many disjoint subsequences can be changed individually so as to flip the label? The formal definition modifies this logic by considering, for each subsequence, not whether changing it to flip the label is possible in principle, but also the probabilities of the different changes. A useful summary statistic is the **average block sensitivity**:

$$\widehat{bs}(f) = \mathop{\mathbb{E}}_{x\sim\Pi}\left[bs(f,x)\right]. \quad (7)$$

Why Consider Subsets? By considering subsets P instead of single indices i, block sensitivity takes into account that words are composed into phrases, and that changing a phrase might change the meaning when changing any individual word cannot. For instance, exchanging the entire phrase 'a gorgeous, witty, seductive' (see Figure 1) with something negative can make the review negative, whereas exchanging any of the individual adjectives cannot, due to the statistical dependencies between the different words. This definition also makes the sensitivity measure robust against tokenization: a more fine-grained tokenization (e.g., into characters) cannot decrease $bs(f,x)$.

3 Sensitivity Bounds for NLP Methods

Many statistical NLP methods proposed over the past decades involve linear combinations of features that look at individual words or groups of a few words. Proposition 1 shows that such methods can only express functions of bounded block sensitivity, with an upper bound quadratic in the number k of inputs the model looks at simultaneously, independently of input length n.

Proposition 1. *Let f be any function $\Sigma^* \to \mathbb{R}$ parameterized as follows:*

$$f(x) := h\left(\frac{1}{n}\sum_{i=1}^{n-k} f_{i,n}(x_i,\ldots,x_{i+k})\right), \quad (x \in \Sigma^n) \quad (8)$$

where $f_{1,n},\ldots,f_{n-k,n}$ are functions $\Sigma^k \to \mathbb{R}^d$ such that $\max_{x\in\Sigma^k} \|f_{i,n}(x)\|_2 \leq C$, and $h : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz continuous. Then, independently of input length n, we have

$$bs(f,x) \leq 2L^2C^2k^2. \quad (9)$$

Proof. Fix a partition $P_1 \dot\cup \ldots \dot\cup P_l = \{1,\ldots,n\}$. Write $g(x)$ for the average inside $h(\cdot)$ in (8). Changing inputs in P_i affects up to $k|P_i|$ of the summands in g. The ℓ_2 norm of the sum of these affected terms is bounded by $\frac{Ck|P_i|}{n}$, and thus

$$\begin{aligned} Var(f|X \in x^{\oplus P_i}) &= \frac{1}{2}\mathop{\mathbb{E}}_{X,Y\in x^{\oplus P_i}} |f(X)-f(Y)|^2 \\ &\leq \frac{1}{2}L^2 \mathop{\mathbb{E}}_{X,Y\in x^{\oplus P_i}} \|g(X)-g(Y)\|_2^2 \\ &\leq 2\frac{L^2C^2k^2|P_i|^2}{n^2}. \end{aligned}$$

Given $\sum_{i=1}^{k}|P_i|^2 \leq (\sum_{i=1}^{k}|P_i|)^2 = n^2$, we find

$$\sum_{i=1}^{l} s(f,x,P_i) \leq 2\frac{L^2C^2k^2}{n^2}\sum_{i=1}^{k}|P_i|^2 \leq 2L^2C^2k^2.$$

□

This result has direct bearing on a wide variety of methods used in NLP, such as averaging word embeddings to construct sentence embeddings (Wieting et al., 2016; Arora et al., 2017; Ethayarajh, 2018), CNNs (Kim, 2014) with average pooling, and log-linear models with n-gram features. The parameter k equals 1 for models averaging word embeddings, the Kernel width for CNNs with average pooling, and n for models using n-gram features. C describes the norm of word embeddings, of the output of a CNN kernel, or of the weights of a linear model. Lipschitz functions h include the sigmoid function σ used in

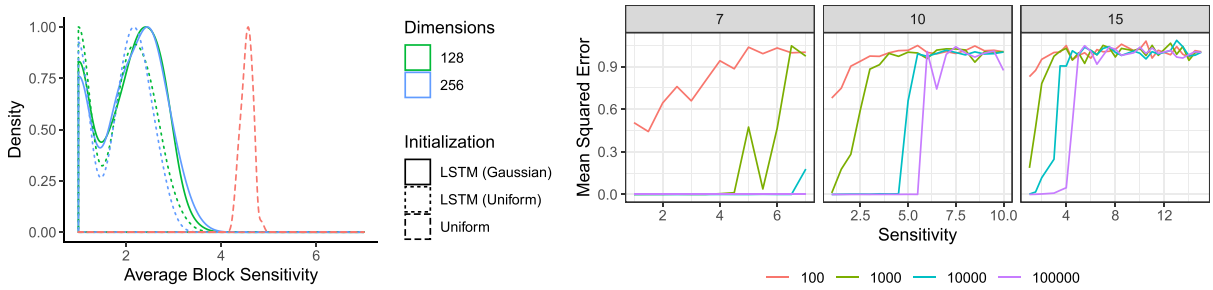

Figure 2: LSTMs are biased towards low-sensitivity functions. (1) *Left*: Distribution of sensitivity of Boolean functions defined by randomly initialized LSTMs (green and blue) and by the uniform distribution (red) over functions $f : \{-1,1\}^7 \to \{-1,1\}$. (2) *Right*: Losses for an LSTM (128 hidden units) fitting random functions $f : \{-1,1\}^N \to \mathbb{R}$ ($N = 7, 10, 15$) with given sensitivities, after 10^2, 10^3, 10^4, 10^5 iterations of training.

logistic regression and its generalization *softmax*, which are 1-Lipschitz, and feedforward networks with Lipschitz activations.

RNNs and LSTMs (Hochreiter and Schmidhuber, 1997) can express functions of any sensitivity, such as f_{Parity}, because they can express all regular languages (Horne and Hush, 1994). On the other hand, transformers (Vaswani et al., 2017) have asymptotically bounded sensitivity as the input length n increases (Hahn, 2020, Lemma 5).

We show that even LSTMs have a learning bias towards low-sensitivity functions, despite their theoretical capacity to represent high-sensitivity functions. We consider functions $f : \{-1,1\}^n \to \mathbb{R}$ where inputs are uniformly distributed over $\{-1,1\}^n$. We first evaluated average block sensitivity both for randomly initialized LSTMs and for the uniform distribution over Boolean functions $\{-1,1\}^7 \to \{-1,1\}$. We constructed Boolean functions from a randomly initialized LSTM by obtaining a scalar output and making this a binary output f based on a threshold chosen to maximize $\text{Var}(f)$. We initialized the LSTM's weights uniformly from $[-d^{-0.5}, d^{-0.5}]$ or from a Gaussian with $\sigma^2 = d^{-1}$, where d is the number of hidden units. Results are shown in Figure 2, for $d = 128$ and $d = 256$. Random Boolean functions have block sensitivity tightly concentrated around ≈ 4.5, whereas the randomly initialized LSTMs consistently show lower block sensitivity. This suggests that low-sensitivity functions are 'overrepresented' in the LSTM parameter space, echoing a theoretical result for feedforward networks (Palma et al., 2019).

Second, we directly examined learnability on functions of different sensitivities. As randomly chosen functions have tightly clustered sensitivity, we sampled[3] functions with a specific targeted average sensitivity $as(f) = \frac{1}{2^n} \sum_{x \in \{-1,1\}^n} s(f,x)$. We did this for sequence lengths $n = 7, 10, 15$. For each $i = 1, \ldots, n$, we constructed five such functions, and then trained an LSTM (128 hidden units) for 10^5 iterations with Adam (learning rate 0.003, batch size 32), and recorded average mean squared error after 10^2, 10^3, 10^4, 10^5 training iterations. Training batches and test examples are sampled uniformly from the 2^n elements of $\{-1,1\}^n$, without consideration of out-of-sample generalization. Results are shown in Figure 2. For $n = 7$, we arrange functions by $\widehat{bs}(f)$; for $n = 10, 15$ we take $as(f)$ instead as it can be computed efficiently and is strongly correlated with $\widehat{bs}(f)$ at $n = 7$ ($R = 0.95$). Low-sensitivity functions are learned perfectly with fewer iterations, whereas high-sensitivity functions are not approximated much better than chance even after 10^5 training iterations. We note that this is a result on the ability to simply *fit* a function of 2^n inputs, not the (harder) task of *generalizing* to unseen input.

4 Sensitivity and Difficulty of NLP Tasks

In Section 3, we provided evidence that sensitivity describes how hard a function is to learn and represent for simple machine learning architectures that do not include pretrained contextual embeddings. In this section, we argue empirically that sensitivity is successful at capturing intuitive notions of task difficulty: Low-sensitivity tasks are those on which simple classifiers as described in Proposition 1, and vanilla LSTMs without

[3]For each $i = 1, \ldots, N$, we sampled functions f where the Fourier spectrum is entirely concentrated on degrees $\{i-1, i, i+1\}$. By O'Donnell (2014, Thm. 2.38), $as(f) \approx i$.

pretraining, are relatively successful. More challenging tasks such as those collected in the GLUE suite (Wang et al., 2019b) have higher sensitivity.

Estimating block sensitivity (5) requires two ingredients: an estimate of the distributions of neighboring strings $\Pi(X|X \in x^{\oplus P})$, and an estimate of f on this set. We approximate Π via a language model, and f via a trained model that is known to attain strong performance on the task. That is, we estimate the sensitivity of a task f by measuring the sensitivity of a model f' that is known to provide close fit to f on the task's input distribution. In Section 5, we report human annotation studies that justify this approximation.

Sampling Neighboring Strings For estimating $\Pi(X|X \in x^{\oplus P})$, we leverage the ability of XLNet (Yang et al., 2019) and u-PMLM (Liao et al., 2020) to model prediction in any order. We use the pretrained `xlnet-large-cased` model provided in Wolf et al. (2019), and a pretrained u-PMLM model trained on the 1 Billion Word benchmark (Chelba et al., 2014). As these models take input on the level of subword tokenizations, we require all samples to consist of the same number of subword symbols as in the span covered by P. To enable meaningful comparison with traditional tokenization and with human intuitions, we only consider subsets P that respect whitespace. We take 10 samples for each P. For tasks with short inputs (text classification and CoLA), we finetune XLNet on the training set to produce completions more in line with the task-specific input distribution. Due to compute availability, we did not apply this procedure to other tasks. Finetuning XLNet slightly increased estimated sensitivity; as we applied it to those tasks expected to have low sensitivity, this procedure potentially makes comparison between tasks more conservative.

Tasks First, we consider four text classification tasks: movie review sentiment (MR, Pang and Lee, 2005), sentence subjectivity (SUBJ, Pang and Lee, 2004), customer reviews sentiment (CR, Hu and Liu, 2004), and opinion polarity (MPQA, Wiebe et al., 2005). On these tasks, low-sensitivity models such as CNNs are known to achieve good performance (Kim, 2014). To approximate the functions f, we finetune `roberta.large.mnli` using `fairseq` for each of the tasks using a single set of hyperparameters.

Second, we selected all tasks of the GLUE challenge suite (Wang et al., 2019b), designed to require a good amount of nontrivial language understading. GLUE contains inference, similarity, and paraphrase tasks (MNLI, Williams et al. (2018); MRPC, Dolan and Brockett (2005); QNLI, Rajpurkar et al. (2016); QQP; STS-B, Cer et al. (2017); RTE, Dagan et al. (2009)), an NLI version of the Winograd schema challenge (Levesque et al., 2012), linguistic acceptability judgments (CoLA, Warstadt et al., 2019), and the Stanford sentiment treebank (SST-2, Socher et al., 2013). On many of these tasks, simple BOW baselines perform essentially at chance (Wang et al., 2019b). We obtain predictions by finetuning RoBERTa (`roberta.large.mnli`) using `fairseq` (Ott et al., 2019) using provided hyperparameters.[4] RoBERTa provides performance close to or exceeding estimated human performance on all GLUE tasks. For the Winograd schema challenge, we took the WSC version from SuperGLUE (Wang et al., 2019a) instead of the NLI reformulation (WNLI) used in GLUE; we used the pretrained model `roberta.large.wsc`. Unlike WNLI, WSC is a single-span task, reducing the number of subsets P considered.

Third, we considered sequence classification formulations of POS tagging and syntactic parsing. For 150 dev sentences in the English Web Treebank (Silveira et al., 2014), we considered the word at the median position of the sentence, and estimated sensitivity of identifying (1) its POS tag in the universal tagset (Petrov et al., 2012), (2) its Universal Dependencies label (Nivre et al., 2016), and (3) the relative position of its head, as an integer. All three tasks are formalized as multiclass classification problems. We estimated all three computations using the pretrained English dependency parser provided in Stanza (Qi et al., 2018; Qi et al., 2020).

Fourth, we considered two datasets probing syntactic knowledge of anaphor licensing (Marvin and Linzen, 2018; Hu et al., 2020), namely, tasks 248 and 260 in SyntaxGym (Gauthier et al., 2020). These tasks ask a model to choose a singular (*himself*) or plural (*themselves*) reflexive after a context where only one is grammatical, but identifying the right reflexive requires syntactic

[4] `https://github.com/pytorch/fairseq/blob/master/examples/roberta/README.glue.md`, retrieved June 1, 2020.

knowledge. We modeled f using the medium-sized GPT2 model (Radford et al., 2019). We chose this task because it could be formalized as binary classification problem, and because GPT2 performed better on this task than on the more familiar subject-verb agreement (and on the feminine version with *herself*).

For each task, we estimated sensitivity for at least 150 dev examples, determined by compute availability. For the syntactic tasks, we estimated sensitivity on the full dataset, as language models are evaluated on these tasks without finetuning.

We considered continuous predictions in $[-1, 1]$ for binary classification tasks, and in $[-1, 1]^d$ for multiclass tasks with d classes, obtained from the sigmoid or softmax layer of the relevant models. For STS-B, we rescale continuous similarity scores to $[-1, 1]$. For parsing and WSC, we used the discrete output labels provided by the pretrained models, represented as one-hot vectors $\in \{-1, 1\}^d$ or binary labels $\in \{-1, 1\}$. For multivariate output $f(x) \in [-1, 1]^d$, we define $s(f, x, P)$ by computing it for each of the d coordinates of $f(x)$, and taking the maximum value over these. The resulting sensitivity estimates describe the behavior of the coordinate of f that has the most nonlinear decision boundary around x.

Lower Bound Approximation Calculating block sensitivity (5) requires calculating the variance for each of the exponentially many subparts P of the input, intractable for all but short inputs. We restrict consideration to a polynomial number of subparts, thus obtaining a *lower bound* on full block sensitivity. We only consider (1) subsets of $1, \ldots, 8$ adjacent tokens, and (2) unions of sets $\{x_{in/7}, \ldots, x_{(i+1)n/7-1}\}$ for $i = 1, \ldots, 7$. For the parsing tasks, we additionally consider all subsets in a window of 7 tokens around the relevant word. This bounds the number of subsets by $8n + 256$, compared to 2^n for full block sensitivity.

4.1 Results

Across the 15 tasks, XLNet and u-PMLM yielded very similar estimates of average block sensitivity ($R = 0.87$, $p = 7 \cdot 10^{-6}$). In Figure 3, we show block sensitivity across tasks as estimated by XLNet. The left panels show kernel density estimates of the distribution over $bs(f, x)$ over the inputs x from the dev sets. The right panels show estimated average block sensitivity $\widehat{bs}(f)$. Text classification tasks have low estimated block

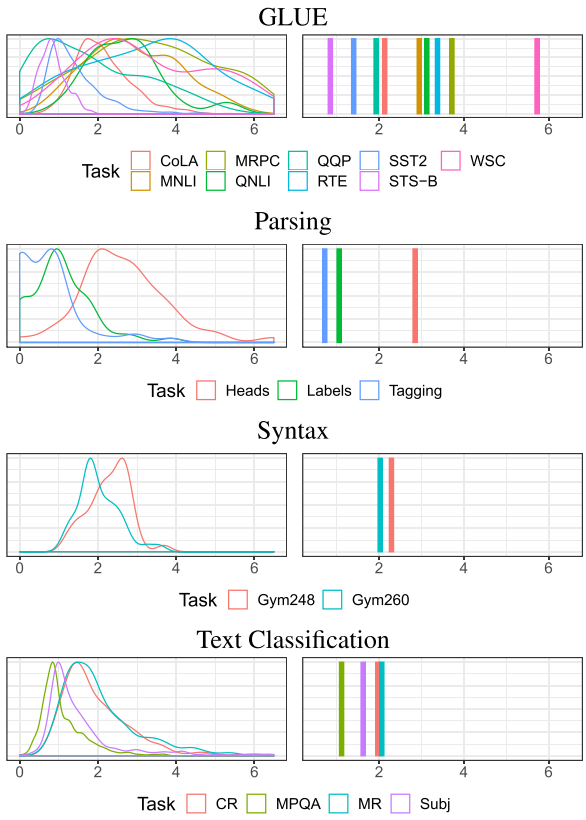

Figure 3: Block sensitivity: For each task, we provide a smoothed histogram of the block sensitivity per input (left), and average block sensitivity (right). Estimates obtained using XLNet; compare Figure 6 for u-PMLM.

sensitivity, with $bs(f, x)$ being concentrated on values lower than three. For the two syntactic tasks, sensitivity is slightly higher; in comparison to the text classification tasks, the histograms show that these tasks have no datapoints with very low sensitivity. For parsing, we see a substantial difference between POS tagging and relation labeling on the one hand, and head identification on the other hand. Identifying tags and relations has lower sensitivity comparable to text classification tasks, whereas identifying the relative position of the head has higher sensitivity. This makes sense: The relative position of the head is sensitive to intervening words that, while not changing the syntactic relation, change the numerical distance between head and dependent. Finally, for GLUE, we observe a wide range of sensitivity scores. SST-2, a sentiment analysis task, has sensitivity very similar to the (other) text classification tasks, as do STS-B (semantic similarity) and QQP (identifying redundant Quora questions). Other tasks show substantially higher scores; the highest estimated average block sensitivities are attained

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

High Sensitivity:

a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior .
2. a painfully bleak ode to bad behavior .
3. a painfully funny ode to bad movies .

Figure 4: Two inputs from SST-2. The first one has low block sensitivity (0.93), as our models find only one sensitive subset P. We show one completion sampled from $x^{\oplus P}$ that flips the label predicted by RoBERTa from POSITIVE to NEGATIVE. The second input has higher block sensitivity (1.88), with three disjoint sensitive subsets. For each subset, we show a completion sampled using XLNet that flips the predicted label.

by RTE, MRPC, and WSC, three tasks designed to require nontrivial reasoning.

To provide insight into these results, we show examples from SST-2 and RTE, with samples from XLNet. In Figure 4, we show two examples from SST-2. The first example has low sensitivity, as our models find only one sensitive subset. On the second example, our models find three disjoint sensitive subsets, leading to higher sensitivity. In Figure 7, we show an example from RTE, consisting of a premise and a hypothesis. The models identify five highly sensitivity subsequences, such that changing the input on any of these subsequences can flip the label from ENTAILMENT to NOENTAILMENT.

Sensitivity and Sentence Length Sensitivity might be higher on longer sentences, because they can be partitioned into more sets P. Does this explain away the differences between tasks? Figure 5 shows per-sentence sensitivity (estimated using XLNet) as a function of sentence length. The left panel compares sensitivity on simple text classification tasks and on CoLA, a GLUE task consisting of short sentences. For the simple text classification tasks, sensitivity increases sharply for very short sentences, but then plateaus. For CoLA, it increases with length. The right panel shows averaged values $bs(f, x)$ across the tasks in each of the four categories. Again, sensitivity increases for GLUE and dependency parsing, while it plateaus for text classification. The two

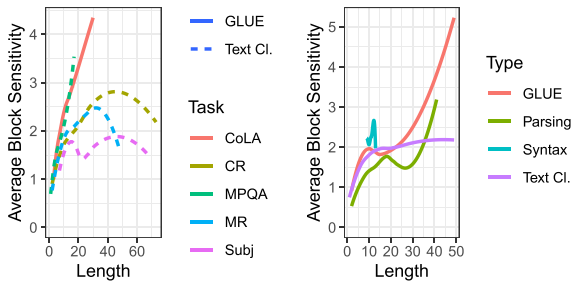

Figure 5: Per-example block sensitivity as a function of sentence length. *Left:* Comparing text classification tasks with CoLA, a single-span GLUE task. *Right:* Block sensitivity across task groups.

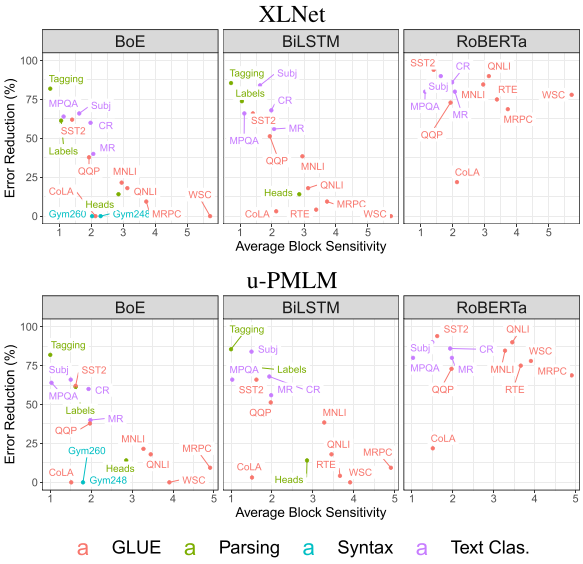

Figure 6: Sensitivity and simple models: Average block sensitivity as estimated using XLNet (top) and u-PMLM (bottom) against error reduction (in % of previously misclassified examples) of a Bag-of-Embeddings (BoE) model, a vanilla BiLSTM, and RoBERTa against the majority class baseline on the dev set.

syntactic tasks consist of short and tightly controlled sentences; in relation to their lengths, their sensitivities are particularly high.

Average Block Sensitivity and Simple Models Based on Section 3, we hypothesized that tasks with low sensitivity correspond to those for which bag-of-words models can meaningfully outperform the majority class baseline, and those on which vanilla LSTM models do best. In Figure 6, we plot average block sensitivity against error reduction (in % of previously misclassified examples) of a bag-of-embeddings (BoE) model,[5]

[5]This model averages GloVE (Pennington et al., 2014) embeddings and applies a one-layer MLP to derive a

Premise:
Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
2. Steve Jobs was attacked by Sculley and the other executives [...] and resigned from the company a few weeks later.

Hypothesis:
Steve Jobs worked for Apple.
3. Jobs later worked for Apple
4. Steve Jobs returned to Apple
5. Steve Jobs worked for Google

Figure 7: An example from RTE, consisting of a premise and a hypothesis. In this example, the premise entails the hypothesis. We show sensitive subsets P_i identified by the models; for each of them, we show one of those completions created by XLNet that flip the label predicted by RoBERTa from ENTAILMENT to NOENTAILMENT. In this example, five highly sensitive subsequences (two in the premise and three in the hypothesis) were identified.

a vanilla BiLSTM,[6] and RoBERTa against the majority class baseline, on the development sets. BoE instantiates the model described in Proposition 1 with $k = 1$; thus, we expect the top right of this graph to be empty for BoE: There can be no high-sensitivity task on which the BoE model provides strong quantitative performance. For both BoE and the vanilla BiLSTM, average sensitivity was negatively associated with error reduction (XLNet: $R = -0.71$, $p = 0.001$ for BoE; $R = -0.82$, $p = 0.0002$ for BiLSTM. u-PMLM: $R = -0.66$, $p = 0.005$ for BoE; $R = -0.76$, $p = 0.002$ for BiLSTM), while no association was observed for RoBERTa (XLNet: $R = -0.05$, $p = 0.87$; u-PMLM: $R = -0.07$, $p = 0.84$). We compared sensitivity as a predictor with label entropy, which showed little association with error reduction of either BoE or the vanilla BiLSTM (both $p > 0.1$).

Which Inputs Have High Sensitivity? We used the Stanford Sentiment Treebank (SST-2, Socher et al., 2013) to investigate which inputs have high sensitivity in sentiment classification. We extracted the 445 dev inputs for which we had estimated sensitivity (determined by compute availability). The dataset contains syntactic parses, with human sentiment annotation for each constituent. We hypothesized that inputs have high sensitivity when different constituents have different sentiment. We focus on estimates from

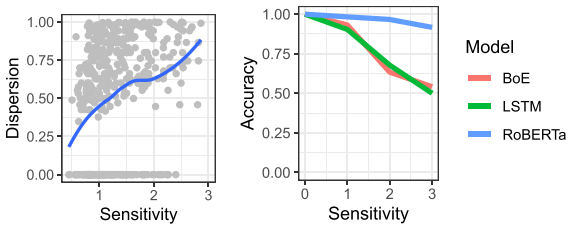

Figure 8: *Left:* Block sensitivity and dispersion (see text) of sentiment labels of constituents. *Right:* Accuracy as a function of sensitivity in sentiment analysis.

XLNet for simplicity; results from u-PMLM are qualitatively identical. We measured the dispersion of sentiment labels over constituents by enumerating positive ($+1$) and negative (-1) labels of all constituents, and computing the standard deviation of this resulting distribution; this is 1 if as many constituents have positive sentiment as there are constituents with negative sentiment. Figure 8 (left) shows this dispersion measure as a function of sensitivity. High-sensitivity examples have higher dispersion. In a linear regression with dispersion and sentence length as predictors of sensitivity, dispersion was highly significant ($\beta = 0.53$, $p < 1.95 \cdot 10^{-10}$), while length was not ($\beta = -0.00$, $p = 0.49$). This is illustrated by the examples in Figure 4 discussed above, where dispersion correlates with sensitivity: The first example has low block sensitivity (0.93) and low label dispersion (0.0); the sentence is labeled positive and no constituent is labeled negative. The second example has higher block sensitivity (1.88) and very high label dispersion (0.94): while the sentence is labeled positive, three constituents are labeled positive and five negative.

Second, we hypothesized that a BoE classifier and a vanilla LSTM perform better on

prediction. This model is called CBOW in Wang et al. (2019b); however, we apply BoE to concatenated spans in the case of multi-span tasks, in line with the definition of sensitivity.

[6]The syntax tasks have no training sets and we thus do not report BiLSTM results; we deduced necessarily at-chance performance for BoE from the design of the task. We excluded STS-B because it cannot be evaluated with accuracy.

low-sensitivity examples, whereas RoBERTa should provide better performance also on higher-sensitivity examples. This is confirmed by Figure 8 (right), where we show the accuracy of BoE, BiLSTM, and RoBERTa as a function of sensitivity. In a logistic regression with sensitivity and sentence length as predictors of BoE accuracy, sensitivity was again highly significant ($\beta = -1.22, p = 4.1 \cdot 10^{-10}$). Findings were similar for the BiLSTM ($\beta = -1.16, p = 1.41 \cdot 10^{-9}$). When predicting the accuracy of RoBERTa, there was still a measurable effect of sensitivity ($\beta = -1.37, p = 1.6 \cdot 10^{-5}$), but overall Figure 8 shows that RoBERTa provides more accurate predictions on higher-sensitivity input. Sentence length was not a significant predictor for accuracy of any of the three models (all $p > 0.05$).

If we choose $s(f, x)$ instead of $bs(f, x)$, namely, restricting to singletons P, there is still a significant effect of $s(f, x)$ on BoE accuracy ($\beta = -1.24, p = 1.1 \cdot 10^{-6}$), but with inferior model fit compared to $bs(f, x)$ (ΔDeviance $= 20.0$), confirming block sensitivity as the more appropriate difficulty measure for simple NLP models.

Role of Task Model We have estimated sensitivity of GLUE and text classification tasks using a large pretrained transformer model (RoBERTa). What would happen if we used a model outside of the family of massive pretrained contextual embeddings? To answer this, we estimated $bs(f, x)$ on SST-2 and RTE using the vanilla BiLSTM to represent f. On SST-2, sensitivity estimated with the BiLSTM's correlated with sensitivty estimated with RoBERTa on those inputs where the BiLSTM provides correct predictions ($R = 0.36, p = 2 \cdot 10^{-11}$), but not on those (typically higher-sensitivity ones) where its predictions are incorrect ($R = 0.15$, $p = 0.21$); a linear regression confirmed that RoBERTa's sensitivity was more predictive of the BiLSTM's sensitivity in those cases that the LSTM labeled correctly ($\beta = 0.2$, $p = 0.004$). On RTE (where the BiLSTM's accuracy is at chance), the BiLSTM's sensitivity was at a constant low value (≈ 0.5) for all inputs. This illustrates that automatic estimation of sensitivity requires a strong model that is able to achieve the sensitivity levels required by a task.

Role of Lower Bound Approximation We evaluated the role of the lower bound approximation on 20 inputs from SST-2 of between 8 and 11 words each—long enough to make the approximation inexact but still allowing consideration of all 2^n subsets. We compared estimates of $bs(f, x)$ based on the approximation (≤ 216 subsets) and the full power set ($\leq 2^{11} = 2048$ subsets). On average, the approximation decreased estimates of $b(f, x)$ from 1.59 to 1.35. However, the two estimates were almost perfectly correlated ($R = 0.95$, $p < 10^{-10}$). Even when restricting to singletons P (up to 11 subsets), the correlation remained high ($R = 0.81$, $p < 0.0001$). Thus, while the approximation may underestimate the numerical values of $bs(f, x)$, it preserves the relative sensitivities of different inputs.

5 Human Validation

In Section 4, we estimated the sensitivity of NLP tasks by plugging a model f' of the task f into equation 5. This methodology requires that the model f' provides good labels on the samples from $x^{\oplus P}$ obtained using the language models. As the language models only approximate the input distribution, their samples could fall outside of the data distribution on which f' approximates the true task f at high accuracy. If this were the case, high estimated sensitivity on tasks such as RTE might reflect brittleness of large models rather than true high sensitivity. Here, we show that this is not the case: Reasoning tasks like RTE have higher sensitivity than text classification tasks like SST-2, even when using human labels.

5.1 Experiment 1: Validating Oracle Model

For 60 items from SST2 and 30 items from RTE each, we collected the subsets $P_1, \ldots, P_k$ achieving the maximum in (5), with 6 samples from XLNet for each subset (we collected fewer items from RTE because they typically have more sensitive subsets P_i, making annotation more expensive). We then recruited naive participants who labeled these samples; each sample was labeled by two or three annotators. In addition to the appropriate labels (''positive'' and ''negative'' for SST-2, ''entails'' and ''does not entail'' for RTE), participants were also provided with a ''makes no sense'' option. We repeated the study for SST2 both with and without finetuning.

The rate of ''makes no sense'' responses on SST-2 was 18% without finetuning and 11% with finetuning; it was 12% on RTE. The agreement between RoBERTa and the modal human

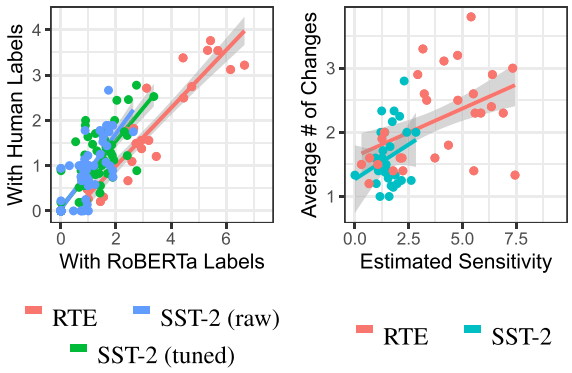

Figure 9: Results of Experiments 1 and 2: *Left:* Sensitivity on SST-2 and RTE calculated using RoBERTa's labels (x-axis) and using human labels (y-axis). On both tasks, both versions are highly correlated ($R > 0.8$ in both tasks). *Right:* Results of Experiment 2: Average number of disjoint subsets on which participants change inputs to flip the label, as a function of estimated sensitivity on SST-2 and RTE.

label was 80% (without finetuning) and 85% (with finetuning) on SST-2, and 72% on RTE; compared to 87%, 92%, and 79%, respectively, average agreement between a single annotator and the modal label. Interannotator agreement is below the human accuracies reported by Nangia and Bowman (2019); we note that the creators of RTE specifically excluded items where human annotators did not agree (Dagan et al., 2009) and that SST-2 excludes reviews labeled as neutral (Socher et al., 2013); we thus expect lower agreement on other strings from the same domain.

The key question is whether these levels of agreement guarantee consistent sensitivity estimates. Figure 9 (left) compares block sensitivity estimated using RoBERTa with values obtained by plugging in average human labels for the function $f(\cdot)$. On both SST-2 and RTE, the values are strongly correlated (SST-2 with and without finetuning both $R = 0.85$; RTE: $R = 0.91$; all $p < 2.2 \cdot 10^{-16}$). On RTE, human estimates are numerically lower than automatic estimates, but the difference in average sensitivity between SST-2 and RTE was strongly replicated by the human estimates ($\beta = 1.3$, $p = 1.3 \cdot 10^{-14}$ in a linear regression). These results indicate that a strong model of a task leads to results similar to a human oracle. In particular, the qualitative difference in sensitivity between SST2 and RTE is replicated when using human labels.

5.2 Experiment 2: Manual Approximation

Experiment 1 showed that human and model labels yield similar results in estimating sensitivity. However, we still relied on the subsets P_i generated by the models. Here, we show that sensitivity, both on the level of individual inputs and on the level of tasks, relates to human intuitions about the number of disjoint subsequences that can be changed to flip the label, which can be easily estimated without any model.

We asked 30 naive individuals to find disjoint subsets in inputs from SST-2 and RTE such that changing the words in any one of them would flip the label. Each participant worked on 30 items from one of the tasks. They rewrote sentences by clicking on words they wanted to replace and entering text replacing those. After submitting a rewrite, participants had the option of identifying another subset disjoint from the previously selected words. They changed at least one subset for every input, and were provided a bonus for every additional subset, incentivizing them to find as many disjoint subsequences as possible. For both SST-2 and RTE, participants were shown an example with instructions guiding them to change three disjoint subsequences. For RTE, we only allowed participants to modify the premise, as similar changes are often possible in premise and hypothesis.

We interpreted the number of disjoint subsets found by participants as a proxy for block sensitivity. This quantity is different from block sensitivity (5), as it does not weight the relative probabilities of all possible changes, and we thus do not expect the same numerical values for both quantities. An exact human estimate of block sensitivity would rely on asking humans both to create multiple samples from $x^{\oplus P}$ for different subsets P and to then label these, infeasible given the large number of possible subsets of each input. In contrast, the task described here only requires annotation from a few annotators for every input.

Figure 9 (right) shows the average number of changes made on each input, as a function of the sensitivity estimated by XLNet+RoBERTa. We conducted a mixed-effects Poisson regression of the number of changes made on the inputs, with random effects for items and subjects. Sensitivity predicted the number of changes ($\beta = 0.061$, $SE = 0.02$, $p = 0.0023$), and there were overall more changes for RTE than for SST2 ($\beta = 0.39$,

$SE = 0.097$, $p = 6 \cdot 10^{-5}$). Input length was not predictive ($\beta = -0.0015$, $SE = 0.002$, $p = 0.32$). This result shows that a fully manual annotation task can approximate differences in sensitivity both between inputs (effect of sensitivity) and between tasks (effects of the contrast between RTE and SST-2).

6 Discussion

We have proposed sensitivity as a theoretical framework for studying the complexity of sequence classification tasks, arguing that it captures complexity both across several machine learning architectures (Section 3) and across NLP tasks (Section 4).

Prior work has studied the ability of RNNs and transformers to represent and learn languages in different classes of the Chomsky hierarchy (e.g., Merrill, 2019). Sensitivity is orthogonal to the Chomsky hierarchy: The maximally sensitive function f_{Parity} has a two-state finite automaton, but there are also low-sensitivity functions that are not even computable. Sensitivity is also distinct from Kolmogorov complexity and similar description length measures (Li and Vitányi, 1993): f_{Parity} has high sensitivity but very low description length. Whereas Kolmogorov complexity is uncomputable and can only be approximated asymptotically, sensitivity can be calculated for individual inputs, enabling us to explicitly evaluate it as a predictor of difficulty on NLP tasks.

Implications for NLP Practice Our results in Section 4 suggest that pretrained contextualized embeddings have been so successful in NLP because they make it possible to learn high-sensitivity functions with modest amounts of task-specific training data. We conjecture that, through large-scale pretraining, models implicitly learn high-sensitivity operations that are generally useful for language understanding. Finetuning such models for classification tasks (Howard and Ruder, 2018; Peters et al., 2018; Devlin et al., 2019) amounts to composing a high-sensitivity model with a low-sensitivity classifier. Some classical techniques can also be interpreted in this light, such as aligning parse trees (a potentially high-sensitivity computation) and extracting features from these alignments that then are fed into an SVM (a low-sensitivity classifier) as an approach to tasks like RTE (Dagan et al., 2009).

Decision Boundaries in NLP The decision boundaries of NLP models are commonly studied to understand their linguistic knowledge (e.g., Linzen et al., 2016; Marvin and Linzen, 2018; Futrell et al., 2019; Jeretic et al., 2020). Kaushik et al. (2020) and Gardner et al. (2020) propose to improve NLP models and their evaluation by specifically considering input pairs that differ in some part and in their (true) label. Dattan et al. (2020) propose to quantify the difficulty of an input by the largest eigenvalue of the Fisher information matrix of a task model, finding that it predicts how sensitive classifiers are to word substitutions.

Sensitivity is different from widely studied pheh nomena of adversarial brittleness (Szegedy et al., 2014; Jia and Liang, 2017): The existence of adversarial examples typically means that natural examples have *some* neighbors, possibly *outside* of the input distribution, on which model output changes even though the true label does not. In contrast, high sensitivity means that there are *many* neighboring inputs *within* the data distribution on which the *true label* changes. Sensitivity may be related to the observation that models often rely on spurious statistical patterns, such as simple lexical correlates of the label observed in reading comprehension datasets (e.g., Kaushik and Lipton, 2018; Gururangan et al., 2018); we expect that such artifacts decrease task sensitivity as they make the gold labels correlated with the output of simple lexical classifiers. Similarly, if the premise alone is predictive of the label in an entailment task (Poliak et al., 2018), changing the hypothesis while staying within the task distribution is less likely to flip the label, again decreasing sensitivity.

Inductive Biases in Neural Networks There is empirical and theoretical evidence that the generalization capabilities of neural networks are in part due to a bias towards ''simple'' functions, with different formal notions of simplicity (e.g., Franco, 2006; Palma et al., 2019; Valle-Perez et al., 2019). A few studies explicitly propose notions similar to low sensitivity as describing simplicity (Franco, 2006; Palma et al., 2019; Novak et al., 2018). Relatedly, empirical work shows that neural networks learn low frequencies in the Fourier spectrum of functions first (Rahaman et al., 2019; Xu et al., 2019; Cao et al., 2019). As low average sensitivity corresponds to concentration of Fourier spectrum on low

frequencies (O'Donnell, 2014, Prop. 3.2), this can be understood as a bias towards low sensitivity. One aspect distinguishing our results here from these prior studies is that we measure sensitivity of realistic functions arising as NLP tasks and on distributions reflecting the nontrivial statistics of natural language. Measuring sensitivity or Fourier spectra on other machine learning tasks is an interesting problem for future research.

7 Conclusion

We proposed block sensitivity as a complexity measure for functions from sequences to labels, applying the measure to quantify the complexity of sequence classification tasks in NLP. Block sensitivity generalizes well-understood complexity measures from the theory of Boolean functions to the setting of natural language. We showed both theoretically and empirically that low sensitivity characterizes tasks on which simple models without massive pretraining provide reasonable performance, and that, in such tasks, more difficult inputs correspond to those with high sensitivity. Our results show that pretrained contextual embeddings enable models to learn tasks with higher sensitivity, and suggest designing challenging tasks by maximizing sensitivity.

Acknowledgments

We thank Judith Degen, Kawin Ethayarajh, Mike Frank, Noah Goodman, and the members of the Stanford NLP group for helpful discussion and feedback. We also thank the anonymous TACL reviewers for their insightful feedback that helped improve the paper. We are also grateful to Yi Liao for providing code and models for u-PMLM. This work was supported by NSF grant #1947307 to RF.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR 2017: International Conference on Learning Representations 2017*.

A. Bernasconi. 1996. Sensitivity vs. block sensitivity (an average-case study). *Information Processing Letters*, 59(3):151–157. https://doi.org/10.1016/0020-0190(96)00105-6

Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. 2019. Towards understanding the spectral bias of deep learning. *arXiv preprint arXiv:1912.01198*.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *INTERSPEECH*, pages 2635–2639.

Noam Chomsky. 1956. Three models for the description of language. *IEEE Transactions on Information Theory*, 2(3):113–124. https://doi.org/10.1109/TIT.1956.1056813

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(4). https://doi.org/10.1017/S1351324909990209

Debajyoti Datta, Shashwat Kumar, Laura E. Barnes, and Tom Fletcher. 2020. Geometry matters: Exploring language examples at the decision boundary. *CoRR*, abs/2010.07212.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

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

Kawin Ethayarajh. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 91–100. https://doi.org/10.18653/v1/W18-3012

Leonardo Franco. 2006. Generalization ability of boolean functions implemented in feedforward neural networks. *Neurocomputing*, 70(1): 351–361. https://doi.org/10.1016/j.neucom.2006.01.025

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 32–42, Minneapolis, Minnesota. Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1004

Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating NLP models via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Jon Gauthier, Jennifer Hu, Ethan Wilcox, Peng Qian, and Roger Levy. 2020. SyntaxGym: An online platform for targeted evaluation of language models. In *Proceedings of the Association for Computational Linguistics: System Demonstrations (ACL 2020)*.

Edward Gibson. 1998. Linguistic complexity: Locality of syntactic dependencies. *Cognition*, 68(1):1–76. https://doi.org/10.1016/S0010-0277(98)00034-1

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL HLT 2018: 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 107–112. https://doi.org/10.18653/v1/N18-2017

Michael Hahn. 2020. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171. https://doi.org/10.1162/tacl_a_00306

John T. Hale. 2001. A probabilistic Earley parser as a psycholinguistic model. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics and Language Technologies*, pages 1–8. https://doi.org/10.3115/1073336.1073357

Pooya Hatami, Raghav Kulkarni, and Denis Pankratov. 2010. Variations on the sensitivity conjecture. *Theory of Computing*, 4:1–27.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Bill G. Horne and Don R. Hush. 1994. Bounds on the complexity of recurrent neural network implementations of finite state machines. In *Advances in Neural Information Processing Systems*, pages 359–366.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL 2018: 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 328–339. https://doi.org/10.18653/v1/P18-1031

Jennifer Hu, Sherry Y. Chen, and Roger P. Levy. 2020. A closer look at the performance of neural language models on reflexive anaphor licensing. *Proceedings of the Society for Computation in Linguistics*, 3(1):382–392.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings*

of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 168–177.

Paloma Jeretic, Alex Warstadt, Suvrat Bhooshan, and Adina Williams. 2020. Are natural language inference models IMPPRESsive? Learning IMPlicature and PRESupposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8690–8705. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.768

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. https://doi.org/10.18653/v1/D17-1215

Jeff Kahn, Gil Kalai, and Nathan Linial. 1988. The influence of variables on Boolean functions. In *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science*, pages 68–80. https://doi.org/10.1109/SFCS.1988.21923

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *ICLR 2020: Eighth International Conference on Learning Representations*.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? A critical investigation of popular benchmarks. In *EMNLP 2018: 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015. https://doi.org/10.18653/v1/D18-1546

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. https://doi.org/10.3115/v1/D14-1181

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *KR'12 Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 552–561.

Ming Li and Paul Vitányi. 1993. *An Introduction to Kolmogorov Complexity and its Applications*, Springer.

Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically masked language model capable of autoregressive generation in arbitrary word order. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 263–274. https://doi.org/10.18653/v1/2020.acl-main.24

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535. https://doi.org/10.1162/tacl_a_00115

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. *arXiv preprint arXiv:1808.09031*. https://doi.org/10.18653/v1/D18-1151

William Merrill. 2019. Sequential neural networks as automata. *arXiv preprint arXiv:1906.01615*. https://doi.org/10.18653/v1/W19-3901

Marvin Minsky and Seymour A. Papert. 1969. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press.

Nikita Nangia and Samuel R. Bowman. 2019. Human vs. muppet: A conservative estimate of human performance on the glue benchmark. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4566–4575. https://doi.org/10.18653/v1/P19-1449

Noam Nisan. 1991. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6): 999–1007. https://doi.org/10.1137/0220062

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan T. McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.

Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. Sensitivity and generalization in neural networks: an empirical study. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Ryan O'Donnell. 2014. *Analysis of Boolean Functions*. Cambridge University Press.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–53.

Giacomo De Palma, Bobak Toussi Kiani, and Seth Lloyd. 2019. Random deep neural networks are biased towards simple functions. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1962–1974.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 271–278. https://doi.org/10.3115/1218955.1218990

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124. https://doi.org/10.3115/1219840.1219855

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. https://doi.org/10.3115/v1/D14-1162

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL HLT 2018: 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 2227–2237. https://doi.org/10.18653/v1/N18-1202

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2089–2096.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191. https://doi.org/10.18653/v1/S18-2023

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 101–108. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. 2019. On the spectral bias of neural networks. In *ICML 2019: Thirty-sixth International Conference on Machine Learning*, pages 5301–5310.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. https://doi.org/10.18653/v1/D16-1264

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Guillermo Valle-Perez, Chico Q. Camargo, and Ard Louis. 2019. Deep learning generalizes because the parameter-function map is biased towards simple functions. In *ICLR 2019: 7th International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR 2019: 7th International Conference on Learning Representations*. https://doi.org/10.1162/tacl_a_00290

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL HLT 2018: 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1112–1122. https://doi.org/10.18653/v1/N18-1101

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew.

2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. 2019. Training behavior of deep neural network in frequency domain. In *International Conference on Neural Information Processing*, pages 264–274. Springer. https://doi.org/10.1007/978-3-030-36708-4_22

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS 2019: Thirty-third Conference on Neural Information Processing Systems*, pages 5753–5763.