How Many and Which Training Points Would Need to be Removed to Flip this Prediction?

Jinghan Yang
The University of Hong Kong
and Northeastern University
eciel@connect.hku.hk

Sarthak Jain
AWS AI Labs*
jsarth@amazon.com

Byron C. Wallace
Northeastern University
b.wallace@northeastern.edu

Abstract

We consider the problem of identifying a *minimal subset* of training data $\mathcal{S}_t$ such that if the instances comprising $\mathcal{S}_t$ had been removed prior to training, the categorization of a given test point x_t would have been different. Identifying such a set may be of interest for a few reasons. First, the cardinality of $\mathcal{S}_t$ provides a measure of robustness (if $|\mathcal{S}_t|$ is small for x_t, we might be less confident in the corresponding prediction), which we show is correlated with but complementary to predicted probabilities. Second, interrogation of $\mathcal{S}_t$ may provide a novel mechanism for *contesting* a particular model prediction: If one can make the case that the points in $\mathcal{S}_t$ are wrongly labeled or irrelevant, this may argue for overturning the associated prediction. Identifying $\mathcal{S}_t$ via brute-force is intractable. We propose comparatively fast approximation methods to find $\mathcal{S}_t$ based on *influence functions*, and find that—for simple convex text classification models—these approaches can often successfully identify relatively small sets of training examples which, if removed, would flip the prediction.[1]

1 Introduction

In this work we pose the following problem in the context of binary classification: *For a test point x_t, identify a minimum subset $\mathcal{S}_t$ of training data that one would need to remove in order to flip the prediction $\hat{y}_t$ for x_t*. This subset may be of interest for a few reasons. First, the cardinality k of $\mathcal{S}_t$ captures one measure of the (in)fragility of the prediction $\hat{y}_t$: Small k indicates that a minor change in the training data would have resulted in a different (discrete) prediction for x_t. We later show that this measure is correlated with, but complementary to, predicted probabilities.

Perhaps a more interesting motivation for recovering $\mathcal{S}_t$ is to provide a potential mechanism for *contesting* model predictions (Hirsch et al., 2017; Vaccaro et al., 2019), i.e., to enable individuals to interrogate and dispute automatic determinations that affect them. If removing a small set of training points would have yielded a different prediction, and if one could make the case for excluding these points (e.g., because they seem mislabeled, or reflect systematic labeling biases), this might provide a compelling case to overturn a model prediction. Consider an educator using an automated essay grading system.[2] Assume the system has output a comparatively poor grade for a student, a determination they see as unfair. Contesting the inclusion of a small set of examples ($\mathcal{S}_t$) which, if excluded, would have resulted in a higher grade provides a novel mechanism for disputation.

Naïvely attempting to find $\mathcal{S}_t$ by brute enumeration and re-training would be hopelessly inefficient. We introduce an algorithm for finding such sets efficiently using *influence functions* (Koh and Liang, 2017) which allow us to approximate changes in predictions expected as a result of removing subsets of training data (Koh et al., 2019). We then provide an iterative variant of this method which does a better job of identifying sets $\mathcal{S}_t$.

Across different datasets and models, we find that we are often able to recover subsets $\mathcal{S}_t$ with relatively small cardinality k; i.e., one can often identify a small to medium subset of training data which, if removed, would flip a given prediction. We also find that there are many test points for which models make predictions with high confidence but where k is small.

The **contributions** here include an investigation of the task of identifying minimal training sets to flip particular predictions in the context of text classification, algorithms for this problem, and an empirical evaluation of their performance in the con-

*Work done prior to joining Amazon.

[1]Code and data to reproduce all experiments available at: https://github.com/ecielyang/Smallest_set

[2]We put aside the question of whether using automated approaches in this particular setting is appropriate to begin with (likely not, though this may depend on how it is used).

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2571–2584
May 2-6, 2023

text of binary text classification.[3]

2 Methods

Assume a binary text classification problem. Given a training set $Z^{\text{tr}} = z_1, ..., z_N$, where $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, we aim to estimate the parameters θ of a classification model $f_\theta : \mathcal{X} \to \mathcal{Y}$ to minimize the empirical risk, i.e., loss $\mathcal{L}$ over Z^{tr}: $\hat{\theta} := \text{argmin}_\theta \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(z_i, \theta) + \frac{\lambda}{2}\theta^T\theta$, which we will denote by $R(\theta)$. We assume throughout that R is twice-differentiable and strongly convex in θ, i.e., $H_{\hat{\theta}} := \nabla^2_\theta R(\hat{\theta}) := \frac{1}{N} \sum_{i=1}^{N} \nabla^2_\theta \mathcal{L}(z_i, \hat{\theta}) + \lambda I$ exists and is positive definite. Suppose we removed a subset of k training points $\mathcal{S} \subset Z^{\text{tr}}$ and re-estimated θ, yielding new parameters $\hat{\theta}_{\mathcal{S}}$. Letting $\varepsilon = -\frac{1}{N}$, we can write this as:

$$\hat{\theta}_S = \text{argmin}_{\theta \in \Theta}\{R(\theta) + \varepsilon \sum_{z_i \in S} \mathcal{L}(z_i, \theta)\} \quad (1)$$

In principle, one could remove the points in $\mathcal{S}$ and re-train to find $\hat{\theta}_{\mathcal{S}}$. In practice this is infeasible given the number of potential subsets $\mathcal{S}$. Koh and Liang (2017) provide (relatively) efficient approximations to estimate $\hat{\theta}_{\mathcal{S}}$ when k=1. Subsequent work (Koh et al., 2019) found that this approximation correlates well with the actual empirical effects of removing a *set* of points (where $k > 1$).

Finding influential subsets Given input x_t, we aim to design an approach to efficiently *identify* the smallest subset $\mathcal{S}_t$ of Z^{tr} such that removing these examples prior to training would change $\hat{y}_t$. Prior work (Cook and Weisberg, 1982; Koh and Liang, 2017) derived the influence exerted by a train point i on the *loss* incurred for a test point t as:

$$-\nabla_\theta \mathcal{L}(z_t, \hat{\theta})^\intercal \underbrace{H^{-1}_{\hat{\theta}} \nabla_\theta \mathcal{L}(z_i, \hat{\theta})}_{\Delta_i \theta} \quad (2)$$

Where $\Delta_i\theta$ is the influence of upweighting z_i during training on estimates $\hat{\theta}$ (Cook and Weisberg, 1982). We are interested, however, in identifying points that have a particularly strong effect on a specific observed *prediction*. We therefore modify Equation 2 to estimate the *influence on prediction* (IP), i.e., the change in predicted probability for x_t observed after removing training instance i. This can be expressed as:

$$\Delta_t f_i := -\nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta_i \theta \quad (3)$$

We then approximate the change in prediction on instance t we would anticipate after removing the training subset $\mathcal{S}_t$ from the training data as the sum of the $\Delta_t f_i$ terms for all points $x_i \in \mathcal{S}_t$.

Algorithm 1 describes a method for constructing $\mathcal{S}_t$. We estimate the change in output expected upon removing each instance from the training dataset and assemble these in $\Delta_t f$. We then greedily consider adding these differences (effectively adding points to $\mathcal{S}_t$) until the resultant output is expected to cross the classification threshold (τ); if we exhaust the training dataset without crossing τ, then we have failed to identify a set $\mathcal{S}_t$.

Algorithm 1: A simple method to find a minimal subset to flip a test prediction

Input: f: Model; Z^{tr}: Full training set; $\hat{\theta}$: Parameters estimated Z^{tr}; $\mathcal{L}$: Loss function; x_t: A test point; τ: Classification threshold (e.g., 0.5)

Output: $\mathcal{S}_t$: minimal train subset identified to flip the prediction ($\emptyset$ if unsuccessful)

$H \leftarrow \nabla^2_\theta \mathcal{L}(Z^{\text{tr}}, \hat{\theta})$
$\Delta\theta \leftarrow H^{-1} \nabla_{\hat{\theta}} \mathcal{L}(Z^{\text{tr}}, \theta')$
$\Delta_t f \leftarrow \nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta\theta$
$\hat{y}_t \leftarrow f(x_t) > \tau$ `// Binary prediction`
`// Sort instances (and estimated output differences) in order of the current prediction`
`direction` $\leftarrow \{\uparrow \text{ if } \hat{y}_t \text{ else } \downarrow\}$
`indices` $\leftarrow$ `argsort`$(\Delta_t f, \texttt{direction})$
$\Delta_t f \leftarrow$ `sort`$(\Delta_t f, \texttt{direction})$
for $k = 1 \ldots |Z^{tr}|$ **do**
  $\hat{y}'_t = (f(x_t) + \texttt{sum}(\Delta_t f[:k])) > \tau$
  if $\hat{y}'_t \neq \hat{y}_t$ **then**
    return $Z^{\text{tr}}[\texttt{indices}[:k]]$
return $\emptyset$

Algorithm 1 is simple and relatively fast, but we can improve upon it by *iteratively* identifying smaller subsets $\mathcal{S}_t$ in Algorithm 2. We detail this approach in Appendix Algorithm 2, but describe it briefly as follows.

We start with the entire train set as a "candidate" $\tilde{\mathcal{S}}_t$, and then iteratively attempt to find strict subsets

[3] Recent related work in economics by Broderick et al. (2020) proposed and investigated a similar problem, with a focus on identifying the sensitivity of econometric analyses to removal of small subsets of data. Recent work on *data-modeling* (Ilyas et al., 2022) also considered a variant of this problem (see Section 5).

Models	Algorithm 1	Algorithm 2
Movie reviews		
BoW	247	151
BERT	484	303
Essays		
BoW	352	134
BERT	484	135
Emotion classification		
BoW	500	345
BERT	524	327
Hate speech		
BoW	808	415
BERT	546	239
Tweet sentiment		
BoW	345	177
BERT	858	569

Table 1: The comparison of average on $k = |\mathcal{S}_t|$ values from Algorithm 1 and Algorithm 2 over the subsets of test points x_t for which we were able to successfully identify a set of points $|\mathcal{S}_t|$

Features	Found $\mathcal{S}_t$	Flip successful
Movie reviews		
BoW	78%	78%
BERT	79%	72%
Essays		
BoW	12%	11%
BERT	9%	8%
Emotion classification		
BoW	91%	91%
BERT	83%	71%
Hate speech		
BoW	67%	60%
BERT	53%	44%
Tweet sentiment		
BoW	99%	91%
BERT	90%	68%

Table 2: Percentages of test examples for which Algorithm 2 successfully identified a set $\mathcal{S}_t$ to remove (center) and for which upon removing these instances and retraining the prediction indeed flipped (right).

of this that by themselves would flip the prediction $\hat{y}_t$. On the first pass, this is equivalent to Algorithm 1, after which—if successful—we will have found a candidate set $\tilde{\mathcal{S}}_t$. Here we update parameter estimates θ to approximate "removing" the points in $\tilde{\mathcal{S}}_t$, and then we recompute the approximation of the influence that points in $\tilde{\mathcal{S}}_t$ would have on $\hat{y}_t$ using a single-step Newton approximation. The idea is that after the parameter update this approximation will be more accurate, potentially allowing us to find a smaller $\mathcal{S}_t$. This process continues until we are unable to find a new (smaller) subset.

In sum, this variant of the algorithm attempts to iteratively identify increasingly small subsets $\tilde{\mathcal{S}}_t$ which would, upon removal prior to training, overturn the original prediction $\hat{y}_t$. There is a computational cost to this, because each iteration involves approximating the influence on a particular prediction; this is computationally expensive. This variant therefore trades run-time for (hopefully) more accurate identification of minimal $\mathcal{S}_t$. However, we find that empirically Algorithm 2 ends up running for only 2.3 passes on average (across all experiments). That is, the algorithm adds a scalar to the run-time of Algorithm 1, but often yields considerably smaller $\mathcal{S}_t$. We show the comparison of $|\mathcal{S}_t|$ returned by two algorithms in the Table 1.

3 Experimental Setup

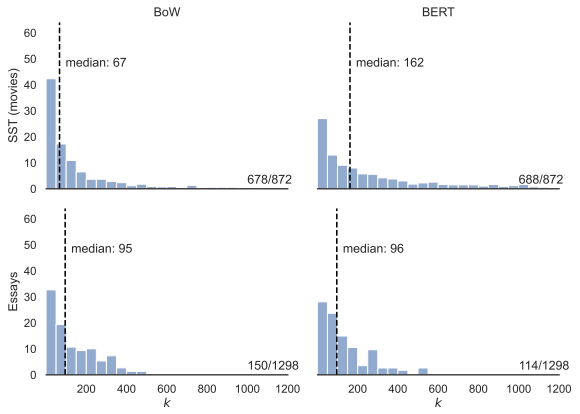
Datasets We use five binary text classification tasks: Movie review sentiment (Socher et al., 2013); Twitter sentiment classification (Go et al., 2009); Essay grading (Foundation, 2010); Emotion classification (Saravia et al., 2018), and; Hate speech detection (de Gibert et al., 2018). We binarize the essay data by labeling the top 10% score points as 1 ("A"s) and others as 0. For the emotion dataset, we include only "joy" and "sadness". We provide dataset statistics in Appendix Table A1. Because the hate speech data is severely imbalanced, we selected a classification threshold τ post-hoc in this case to maximize train set F1 (yielding $\tau = 0.25$); for other datasets we used $\tau = 0.5$, which corresponded to reasonable F1 scores—for reference we report prediction performance on all datasets in Appendix Table A2.

Models We consider only $\ell 2$ regularized logistic regression (for which influence approximation is well-behaved). As features, we consider both bag-of-words and neural embeddings (induced via BERT; Devlin et al. 2018).

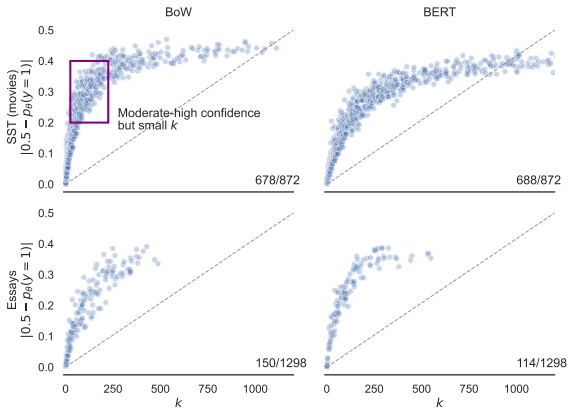
4 Results

Here we present results for the iterative method (Appendix Algorithm 2), which outperforms the simpler Algorithm 1. We provide full results for both methods in the in the Appendix.

How often can we find $\mathcal{S}_t$ and how frequently does removing the instances it contains flip the prediction? As can be seen in Table 2, this varies considerably across datasets. For movie reviews, Algorithm 2 returns a set $\mathcal{S}_t$ for $\sim$80% of test points, whereas for the (more complex) essays data it does so for only $\sim$10% of instances. Other datasets see success somewhere in-between these extremes. However, when the algorithm does return a set $\mathcal{S}_t$, removing this and re-training almost



(a) Histograms of $k = |\mathcal{S}_t|$ values over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$. We report the fraction for which we were able to do so in the lower sub-plot right corners.



(b) Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified. These are correlated (as we would expect), but there are many points for which the model is moderately or highly confident, but where removing a relatively small set of training data would change the prediction.

Figure 1: Results characterizing $\mathcal{S}_t$ on two illustrative datasets (sentiment classification and essay scoring).

always flips the prediction $\hat{y}_t$ (right-most column).

What is the distribution of $k = |\mathcal{S}_t|$? Figure 1a shows empirical distributions of k values for subsets $\mathcal{S}_t$ identified by Algorithm 2 for the illustrative movie review and essay grading datasets (full results in Appendix). The take-away here is that when we do find $\mathcal{S}_t$, its cardinality is often quite small. Indeed, for many test points removing tens of examples would have flipped the prediction.

How does k relate to predicted probability? Does the size of $\mathcal{S}_t$ tell us anything beyond what we might infer from the predicted probability $p(y_t = 1)$? In Figure 1b we show (again for just two datasets here) a scatter of $k = |\mathcal{S}_t|$ against the distance of the predicted probability from 0.5. The former provides complementary information, in that there exist instances about which the model is confident, but where removing a small set of training instances would overturn the prediction.

Qualitative example. One reason to recover sets $\mathcal{S}_t$ is to support *contestation*—if k is small, one might argue against the appropriateness of the points in $\mathcal{S}_t$ and hence against the determination y_t. As a simple example,[4] consider the movie review test instance "*Manages to transcend the sex drugs and show tunes plot into something far richer*". The true label is positive, but the model predicted negative. Algorithm 2 reveals that removing a single example ($k = 1$) from the training set would have reversed the prediction—specifically, this negative review: "*An overstylized pureed melange of sex psychology drugs and philosophy*". It seems this training point is only superficially similar to the test point, which may make a case for overturning the prediction. While standard influence functions (Koh and Liang, 2017) can be used to *rank* training points, the novelty here is observing that *removing this point alone* would change the prediction.

[4]We provide more qualitative analysis in the Appendix.

5 Related Work

Influence functions (Hampel, 1974; Cook and Weisberg, 1980, 1982) provide machinery to identify training points that most informed a particular test prediction. Influence can provide insight into predictions made by modern neural networks (Koh and Liang, 2017), and can be used to *debug* models and training data by surfacing mislabeled training points and/or reliance on artifacts (Adebayo et al., 2020; Han et al., 2020; Pezeshkpour et al., 2022; Teso et al., 2021), and tuning influence can be used to demote reliance on unwanted correlations (Han and Tsvetkov, 2021). Influence can also be used to *audit* models by inspecting training data responsible for predictions Marx et al. (2019).

Schulam and Saria (2019) audit individual predictions by approximating how much they might have changed under different samples from the training distribution. Ting and Brochu (2018) consider influence functions as a tool for optimally subsampling data in service of computational effi-

ciency. Koh et al. (2019) considered approximating the effect of removing a *group* of training points using influence functions, and found that they do so fairly well (a result that we use). They assumed groups were *given* and then evaluated the accuracy of the influence approximation to the change in prediction. By contrast, we are interested in *finding* a (minimal) group which would have the specific effect of flipping a prediction. Elsewhere, Khanna et al. (2019) ask: "Which training examples are most responsible for a given *set* of predictions?".

Broderick et al. (2020) assess the robustness of economic analyses when a fraction of data is removed. They therefore focus on the magnitude/significance of parameter estimates. This framing differs from our ML-centric motivation, which aims to recover specific small subsets of data that, if removed, would change a particular prediction (and so might support contestability).

Robustness of data analyses to dropping training data In the process of review it was brought to our attention that Broderick et al. (2020) addressed a closely related problem to what we have considered here, albeit from a quite different motivating perspective—namely assessing the sensitivity of econometric analyses to removals of small subsets of data. It turns out that the algorithm that was (independently) proposed by Broderick et al. (2020) in that work is similar to Algorithm 1. The present effort is novel in our focus on machine learning, and specifically on identifying minimal subsets of training data which would flip a particular prediction if removed prior to training.

Minimal feature set removal Another related line of work concerns a natural complement to the problem we have considered: Instead of identifying a minimal set of *instances* to remove in order to change a prediction, the idea is to find a minimal subset of *features* such that, if these were set to uninformative values, a particular prediction would change (Harzli et al., 2022). Work on *counterfactual examples* has similarly sought to identify minimal (feature) edits to instances that would change the associated label (Kaushik et al., 2019).

Datamodeling Recent work on *datamodeling* (Ilyas et al., 2022) provided a generalized framework for analyzing model behavior as a function training data. This approach entails *learning to estimate* (via a parameterized model) changes we would anticipate observing for a particular instance if the model had been trained on some subset of the original training set. This approach is flexible, and one thing it permits is identifying the *data support* of a particular prediction for x_t, i.e., what we have called $\mathcal{S}_t$ (4.1.1 in Ilyas et al. 2022). Furthermore, this method is not restricted to the simple regularized linear models we have considered here. However, this comes with the downside of high computation costs: One needs to re-train the algorithm being modeled many times with different training data subsets to yield a "training set" to be used to estimate model behavior under conterfactual training sets. The main comparative advantage of our more focused approach is therefore relative computational efficiency.

Contestability (Vaccaro et al., 2019; Almada, 2019) in ML is the idea that individuals affected by a prediction ought to be able to challenge this determination, which may require parties to "marshal evidence and create counter narratives that argue precisely why they disagree with a conclusion drawn by an AI system" (Hirsch et al., 2017). The right to contestability is in some cases enshrined into law (Almada, 2019). Identifying $\mathcal{S}_t$ for review by an individual affected by the prediction $\hat{y}_t$ may constitute a concrete mechanism for contestation.

6 Conclusions

In the context of binary text classification, we investigated the problem of identifying a minimal set of training points $\mathcal{S}_t$ such that, if excluded from training, the prediction for test instance x_t would flip. We proposed two relatively efficient algorithms for this—both using approximate group influence (Koh and Liang, 2017; Koh et al., 2019)—and showed that for regularized linear models they can often find relatively small $\mathcal{S}_t$. We provided empirical evidence that this captures uncertainty in a way that is somewhat complementary to predicted probabilities, and may serve as a mechanism to support *contestability*, by allowing individuals to review (and dispute) instances in $\mathcal{S}_t$.

Limitations

A key limitation of this work is that we have restricted analysis to regularized linear models with convex loss. We leave extension and evaluation of the proposed methods for more complex models to future work. Indeed, our hope is that this initial effort inspires further work on the problem of identifying minimal train sets which would overturn a

specific prediction if removed.

More conceptually, the implications of finding a small subset $\mathcal{S}_t$ are not entirely clear. Intuitively, small sets would seem to indicate fragility, but we have not formalized or evaluated this further. Moreover, there may in certain cases exist *multiple* (distinct) subsets $\mathcal{S}_t$, such that removing any of these subsets would flip the prediction for x_t. This would complicate the process of contestation envisioned. Furthermore, assuming a stochastic parameter estimation method (e.g., SGD) the composition of $\mathcal{S}_t$ may depend on the arbitrary random seed, similarly complicating the interpretation of such sets.

Acknowledgements

We thank our anonymous EACL reviewers for helpful feedback, especially with respect to relevant related work. We also thank Gautam Kamath for highlighting connections to the *datamodeling* work (Ilyas et al., 2022). This work was supported in part by the Army Research Office (W911NF1810328), and in part by the Overseas Research Fellowship under the University of Hong Kong.

Ethics Statement

Models are increasingly used to make (or aid) decisions that directly affect individuals. In addition to the broader (potential) "interpretability" afforded by recovering small sets of training data that would change a prediction if removed, this may provide a new mechanism for individuals to contest such automated decisions, specifically by disputing this set of training data in some way. However, our proposed method only finds training points that highly impact the model prediction for a given example; these may or may not be noisy or problematic instances. Human judgement is required to assess the accuracy and relevancy of the instances in $\mathcal{S}_t$.

A broader view might be that classification models are simply not appropriate for the kinds of sensitive applications we have used as motivation here. The use of (semi-)automated methods for essay grading, e.g., has long been debated (Hearst, 2000). One might argue that rather than trying to provide mechanisms to contest ML predictions, a better choice may be not to use models in cases where these would be necessary at all. We are sympathetic to this view, but view the "appropriateness" of ML for a given problem as a spectrum; contestability may be useful even in "lower stakes" cases. Moreover, the general problem we have introduced of identifying small training sets which can by themselves swing predictions, and the corresponding methods we have proposed for recovering these, may be of intrinsic interest beyond contestability (e.g., as an additional sort of model uncertainty).

References

Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. 2020. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*.

Marco Almada. 2019. Human intervention in automated decision-making: Toward the construction of contestable systems. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pages 2–11.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR.

Tamara Broderick, Ryan Giordano, and Rachael Meager. 2020. An automatic finite-sample robustness metric: When can dropping a little data make a big difference? *arXiv preprint arXiv:2011.14999*.

Guillaume Charpiat, Nicolas Girard, Loris Felardos, and Yuliya Tarabalka. 2019. Input similarity from the neural network perspective. *Advances in Neural Information Processing Systems*, 32.

R Dennis Cook and Sanford Weisberg. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508.

R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Hewlett Foundation. 2010. The hewlett foundation: Automated essay scoring.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Xiaochuang Han and Yulia Tsvetkov. 2021. Influence tuning: Demoting spurious correlations via instance attribution and instance-driven updates. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4398–4409, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676*.

Ouns El Harzli, Bernardo Cuenca Grau, and Ian Horrocks. 2022. Minimal explanations for neural network predictions. *arXiv preprint arXiv:2205.09901*.

Marti A Hearst. 2000. The debate on automated essay grading. *IEEE Intelligent Systems and their Applications*, 15(5):22–37.

Tad Hirsch, Kritzia Merced, Shrikanth Narayanan, Zac E Imel, and David C Atkins. 2017. Designing contestability: Interaction design, machine learning, and mental health. In *Proceedings of the 2017 Conference on Designing Interactive Systems*, pages 95–99.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. 2022. Datamodels: Understanding predictions with data and data with predictions. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9525–9587. PMLR.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*.

Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. 2019. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390. PMLR.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. 2019. On the accuracy of influence functions for measuring group effects. *Advances in neural information processing systems*, 32.

Charles Marx, Richard Phillips, Sorelle Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. 2019. Disentangling influence: Using disentangled representations to audit model predictions. *Advances in Neural Information Processing Systems*, 32.

Pouya Pezeshkpour, Sarthak Jain, Sameer Singh, and Byron Wallace. 2022. Combining feature and instance attribution to detect artifacts. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1934–1946, Dublin, Ireland. Association for Computational Linguistics.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Peter Schulam and Suchi Saria. 2019. Can you trust this prediction? auditing pointwise reliability after learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1022–1031. PMLR.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y 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.

Stefano Teso, Andrea Bontempelli, Fausto Giunchiglia, and Andrea Passerini. 2021. Interactive label cleaning with example-based explanations. *Advances in Neural Information Processing Systems*, 34:12966–12977.

Daniel Ting and Eric Brochu. 2018. Optimal subsampling with influence functions. *Advances in neural information processing systems*, 31.

Kristen Vaccaro, Karrie Karahalios, Deirdre K Mulligan, Daniel Kluttz, and Tad Hirsch. 2019. Contestability in algorithmic systems. In *Conference Companion Publication of the 2019 on Computer Supported Cooperative Work and Social Computing*, pages 523–527.

Dataset	# Train	# Test	% Pos
Movie reviews	6920	872	0.52
Essays	11678	1298	0.10
Emotion	9025	1003	0.53
Hate speech	9632	1071	0.11
Tweet sentiment	18000	1000	0.50

Table A1: Text classification dataset statistics.

Models	Accuracy	F1-score	AUC
Movie reviews			
BoW	0.79	0.80	0.88
BERT	0.82	0.83	0.91
Essays			
BoW	0.97	0.80	0.99
BERT	0.97	0.84	0.99
Emotion classification			
BoW	0.77	0.79	0.86
BERT	0.80	0.82	0.88
Hate speech			
BoW	0.87	0.40	0.81
BERT	0.89	0.63	0.88
Tweet sentiment			
BoW	0.70	0.70	0.75
BERT	0.75	0.76	0.84

Table A2: The model performance respect to datasets included in the experiment.

A Appendix

A.1 Dataset Statistics and Predictive Performance

We present basic statistics describing our text classification datasets in Table A1. For the tweet sentiment dataset, we randomly sampled 19000 points from the 1600000 points to make experiments feasible. For reference, we also report the predictive performance realized by the models considered on the test sets of these corpora in Table A2.

A.2 Full results

Table A3 reports the percentages of instances for which Algorithm 1 identifies a subset $\mathcal{S}_t$ (center column), and for which this set actually flipped the prediction following removal (right column). Contrast this with Table 2, which reports the same for the proposed iterative approach in Algorithm 2.

We provide histograms of $k = |\mathcal{S}_t|$ for the sets we were able to identify via Algorithm 1 in Figure A.2, and the same plots for Algorithm 2 in Figure A.4.

Finally, we plot the relationship between k and predicted probabilities under Algorithms 1 and 2 in Figures A.3 and A.5, respectively.

Features	Found $\mathcal{S}_t$	Flip successful
Movie reviews		
BoW	78%	78%
BERT	79%	76%
Essays		
BoW	12%	12%
BERT	9%	9%
Emotion classification		
BoW	91%	91%
BERT	83%	78%
Hate speech		
BoW	67%	65%
BERT	53%	49%
Tweet sentiment		
BoW	99%	98%
BERT	90%	73%

Table A3: Percentages of test examples for which Algorithm 1 successfully identified a set $\mathcal{S}_t$ (center) and for which upon removing these instances and retraining the prediction indeed flipped (right).

A.3 Additional qualitative analysis

We conclude with a brief qualitative analysis of examples in $\mathcal{S}_t$ retrieved in the case of the essays data. The model operating over BERT representations classified this test point (x_t) as 0, i.e., not an "A": "*The cyclist in this essay was a very brave man* ...". The example is about a paragraph in length total, but details adventures of a cyclist. In this case it happens that the reference label is, in fact, an "A", so the model is incorrect. Algorithm 2 reveals that removing a single training point and retraining would have overturned this prediction, yielding an "A". The point in question is labeled 0 (so below an "A") and is about the mood of a memoir, in particular arguing that the person central to this was happy. The student-author of the cyclist essay might reasonably argue that this example is not at all relevant to their essay, and the fact that excluding this single example would have meant their essay received an "A" may be an adequate case for changing their grade accordingly.

A.4 Time complexity

We recorded wall clock times required to search for $|\mathcal{S}_t|$ on all test points in each dataset using Algorithm 1 and Algorithm 2 on Intel(R) Core(TM) i9-9920X CPUs; we report these times in Table A4. For Algorithm 1, the longest running time is required for the essay dataset because most test predictions cannot be flipped even after iterating over all training points. Algorithm 2 is considerably slower than Algorithm 1. The main reason lies in recording the set of training points not in $\mathcal{S}_t$ (line 20 in Algorithm 2) and re-calculating the IP

value in each iteration to reduce the minimal candidate set. This additional time is traded off against the ability to (typically) find smaller $\mathcal{S}_t$ compared with 1. Overall, the running time required to find $|\mathcal{S}_t|$ for one test point is relatively minimal for both algorithms.

A.5 Attribution methods

We consider different methods (i.e., other than influence functions) to rank training instances including gradient similarities in terms of the loss, similarity-based methods and randomly sampling training points. Of these we found that the proposed method works best in terms of finding instances which exert maximal influence on the prediction.

One natural way to quantify the impact of a training point x_i on a training point x_t by similarity methods. If the model has training points similar to the test point, it may classify the test correctly with high probability. We consider three of similarity-based methods: **EUC** $= -||x_t - x_i||^2$, **DOT** $= \langle x_i, x_t \rangle$, and **COS** $= cos(x_i, x_t)$.

Apart from influence function and IP, we consider gradient-based instance attribution methods:

1) $RIF = cos(H^{-\frac{1}{2}} \nabla_\theta \mathcal{L}(x_t), H^{-\frac{1}{2}} \nabla_\theta L(x_i))$
2) $GD = \langle \nabla_\mathcal{L} L(x_t), \nabla_\theta L \mathcal{L} \rangle$
3) $GC = cos(\nabla_\theta \mathcal{L}(x_t), \nabla_\theta \mathcal{L}(x_i))$

RIF was proposed to mitigate the issues of outliers and mislabeled points being returned by the standard influence functions (Barshan et al., 2020). GC and GD measure the similarity between two instances can also become an effective way to interpret the model from the instance perspective (Charpiat et al., 2019). Apart from the methods above, we randomly sample training subsets and remove them accordingly.

We apply the above methods to the movie review dataset trained with a logistic regression model. We evaluate each attribution method as follows: First, we remove the top $k = |\mathcal{S}_t|$ training points from the training dataset according to the score calculated from the attribution method. Then we train a new with the same dataset except for the removed points. Finally, we compare the difference in predictions for each test point from the old model to the new model. To show the impact of attribution methods under different $k = |\mathcal{S}_t|$, we iterated with $k = |\mathcal{S}_t|$ from 50 to 3000. The mean absolute difference is plotted along with $k = |\mathcal{S}_t|$ shown in Figure A.1. IP has a larger impact on the predicted probability, compared to removing training points ranked according to other methods.

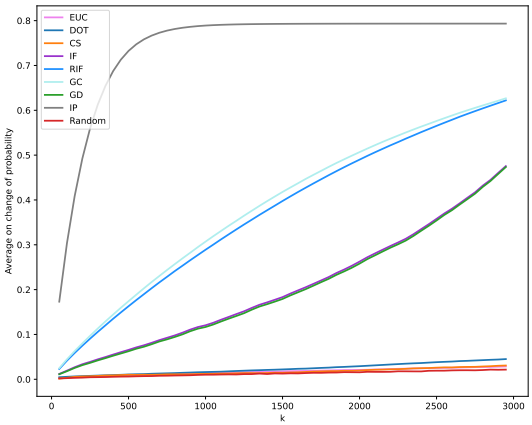


Figure A.1: The relationship between the mean of absolute difference on predicted probabilities for all test points results from removing $|\mathcal{S}_t|$ training points, using different methods.

Datasets		Movie reviews	Essays	Emotion	Hate speech	Tweet
Bow	Algorithm 1	5	155	5	40	11
	Algorithm 2	239	257	534	444	1529
BERT	Algorithm 1	3	161	19	52	8
	Algorithm 2	604	288	761	522	2203

Table A4: Running time (in seconds) to find $|\mathcal{S}_t|$ for all test points in each data set by Algorithm 1 and Algorithm 2.

Algorithm 2: An *iterative* approach to finding a minimal set to flip a prediction

Input: f: Model; Z^{tr}: Full training set; $\hat{\theta}$: Parameters estimated Z^{tr}; $\mathcal{L}$: Loss function; x_t: A test point; τ: Classification threshold (e.g., 0.5)

Output: $\mathcal{S}_t$: minimal train subset identified to flip prediction for x_t ($\emptyset$ if unsuccessful)

$\theta' \leftarrow \hat{\theta}$

$Z^{\text{tr}}_{\text{r}}, \tilde{\mathcal{S}}_t \leftarrow Z^{\text{tr}}, Z^{\text{tr}}$ `// Track remaining points and candidate subset` $\tilde{\mathcal{S}}_t$

$H \leftarrow \nabla^2_\theta \mathcal{L}(Z^{\text{tr}}, \theta')$

$\Delta\theta \leftarrow H^{-1} \nabla_\theta \mathcal{L}(Z^{\text{tr}}, \theta')$

$\Delta_t f \leftarrow \nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta\theta$

$\hat{y}_t \leftarrow f(x_t) > \tau$

$\Delta_t f_{\text{sum}} \leftarrow 0$

$k' \leftarrow |Z^{\text{tr}}|$

while $\tilde{\mathcal{S}}$ *changed since last iteration* **do**

  `// Sort instances (and estimated output differences) in order of the current prediction`

  $\texttt{direction} \leftarrow \{\uparrow \text{ if } \hat{y}_t \text{ else } \downarrow\}$

  $\texttt{indices} \leftarrow \texttt{argsort}(\Delta_t f, \texttt{direction})$

  $\Delta_t f \leftarrow \texttt{sort}(\Delta_t f, \texttt{direction})$

  for $k = 1 \ldots |\tilde{\mathcal{S}}_t|$ **do**

    $\hat{y}'_t \leftarrow (f(x_t) + \texttt{sum}(\Delta_t f[:k])) > \tau$

    if $\hat{y}'_t \neq \hat{y}_t$ **then**

      $\Delta_t f_{\text{sum}} \leftarrow \texttt{sum}(\Delta_t f[:k])$

      $\texttt{diff} \leftarrow k' - k$

      $k' \leftarrow k$

      $\tilde{\mathcal{S}}_t \leftarrow \tilde{\mathcal{S}}_t[\texttt{indices}[:k]]$ `// Update candidate subset`

      $Z^{\text{tr}}_{\text{r}} \leftarrow Z^{\text{tr}} / \tilde{\mathcal{S}}_t$ `// And the set of training points not in` $\tilde{\mathcal{S}}_t$

      $\theta' \leftarrow \theta + \Delta\theta[\texttt{indices}[:k]]$

      `// Update Hessian and` ∇ `of loss using updated` θ `estimate`

      $H \leftarrow \nabla^2_\theta \mathcal{L}(Z^{\text{tr}}_{\text{r}}, \theta')$

      $\Delta\theta \leftarrow H^{-1} \nabla_\theta \mathcal{L}(\tilde{\mathcal{S}}_t, \theta')$

      $\Delta_t f \leftarrow \nabla_\theta f_{\hat{\theta}}(x_t)^\intercal \Delta\theta$

      `break`

if $|\tilde{\mathcal{S}}_t| = |Z^{tr}|$ **then**

  return $\emptyset$

return $\tilde{S}_t$

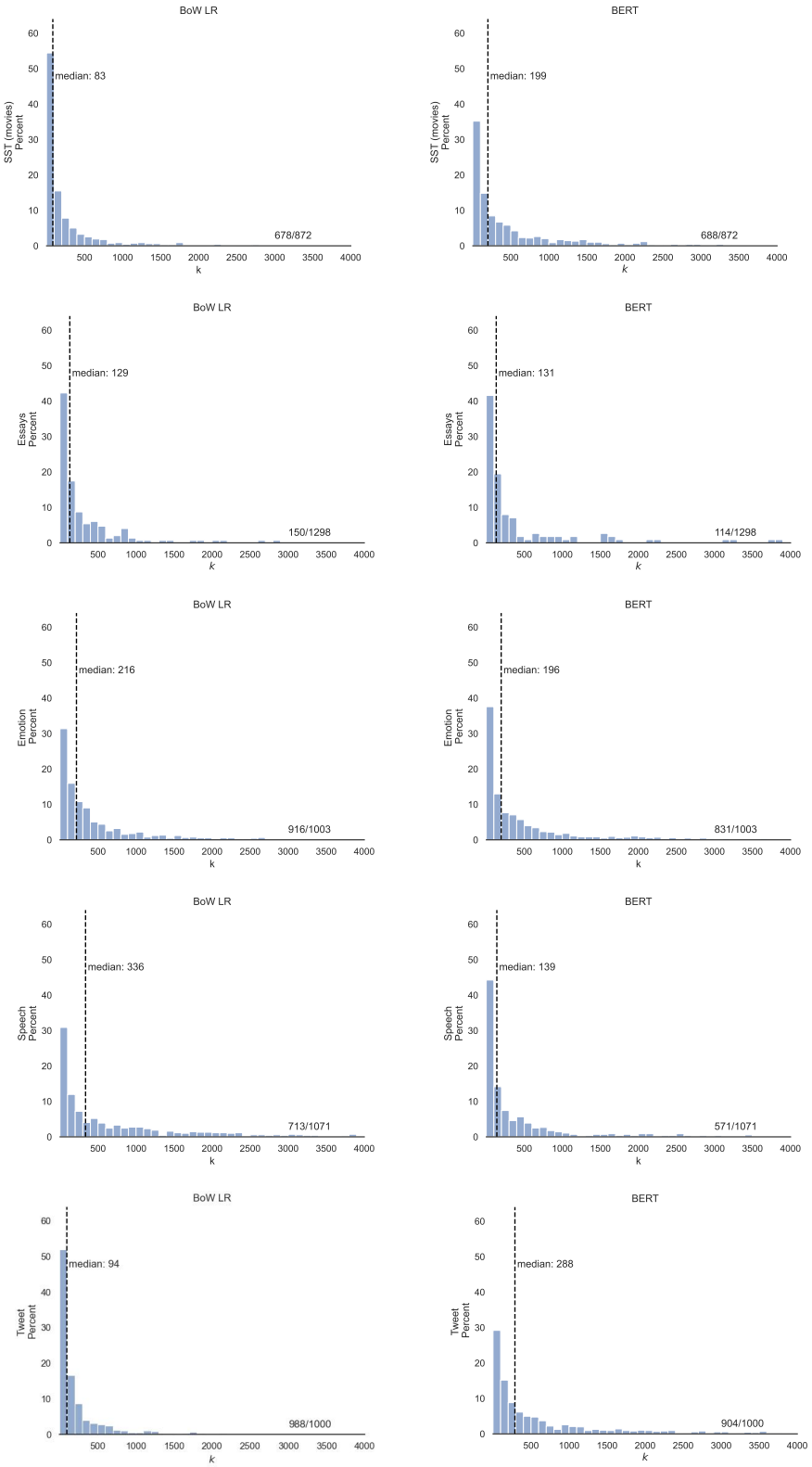


Figure A.2: Histograms of $k = |\mathcal{S}_t|$ values from Algorithm 1 over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$.

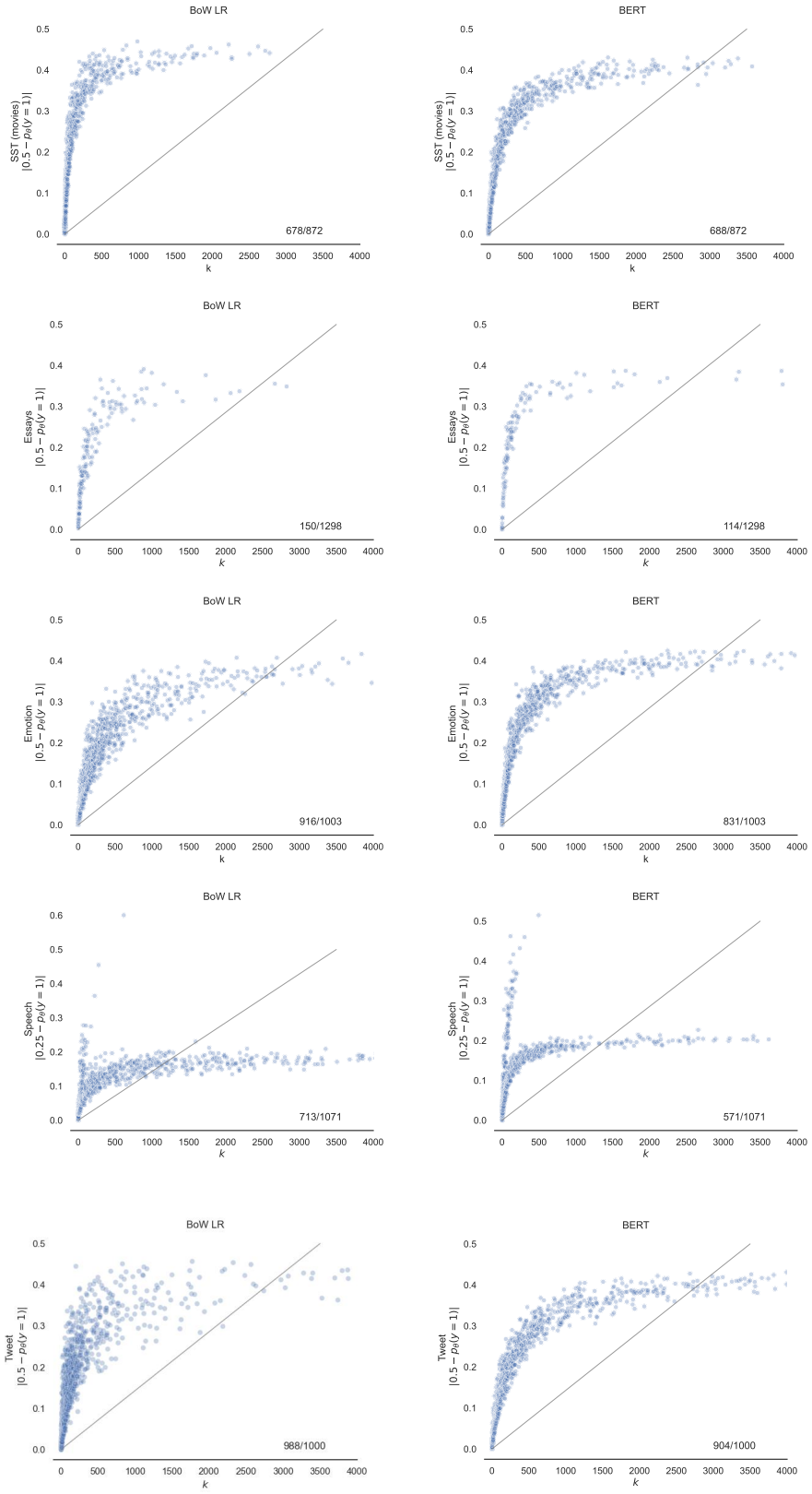


Figure A.3: Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified from Algorithm 1.

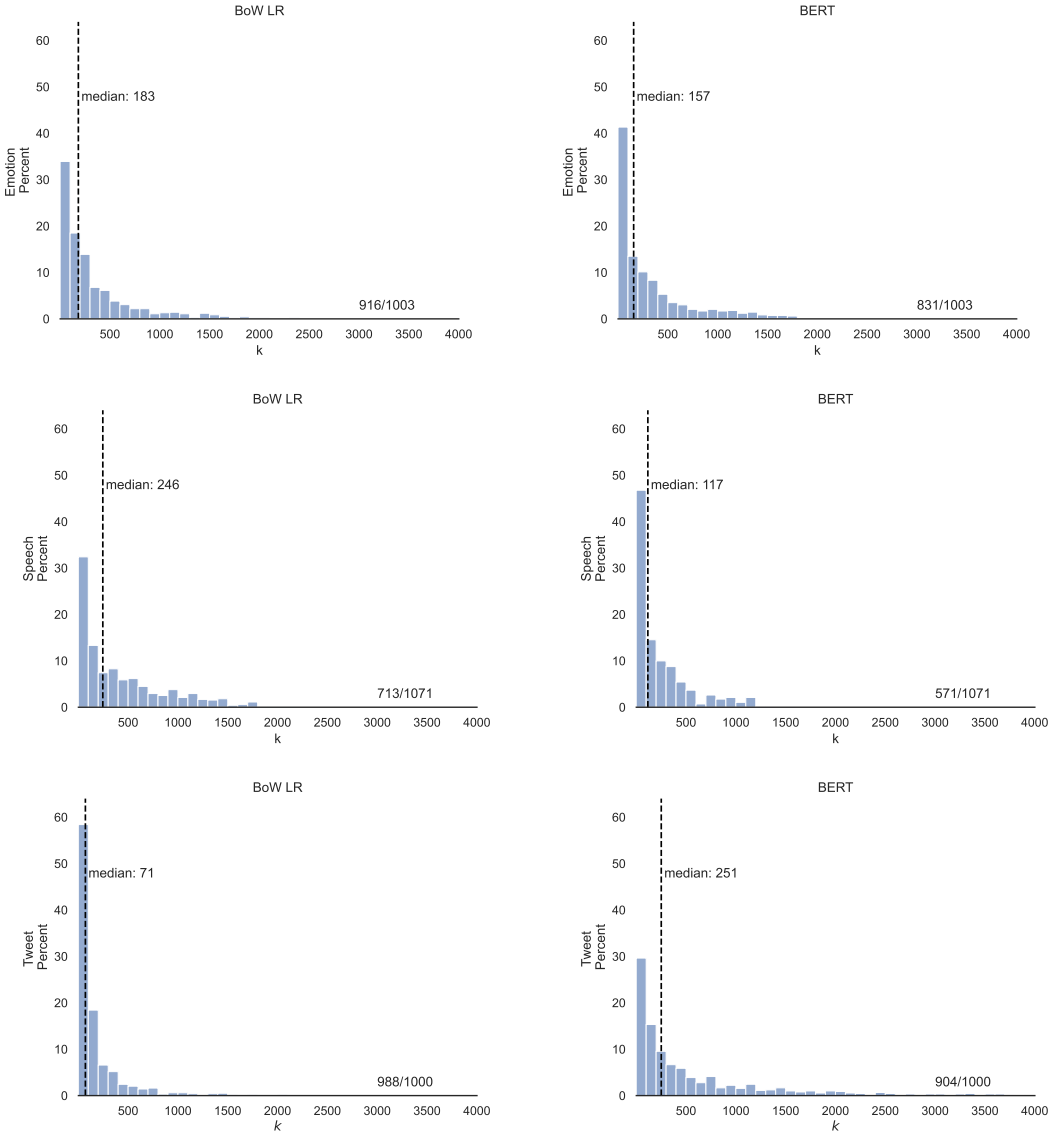


Figure A.4: Histograms of $k = |\mathcal{S}_t|$ values from Algorithm 2 over the subsets of test points x_t for which we were able to successfully identify a set of points $\mathcal{S}_t$ such that removing them would flip the prediction for $\hat{y}_t$.

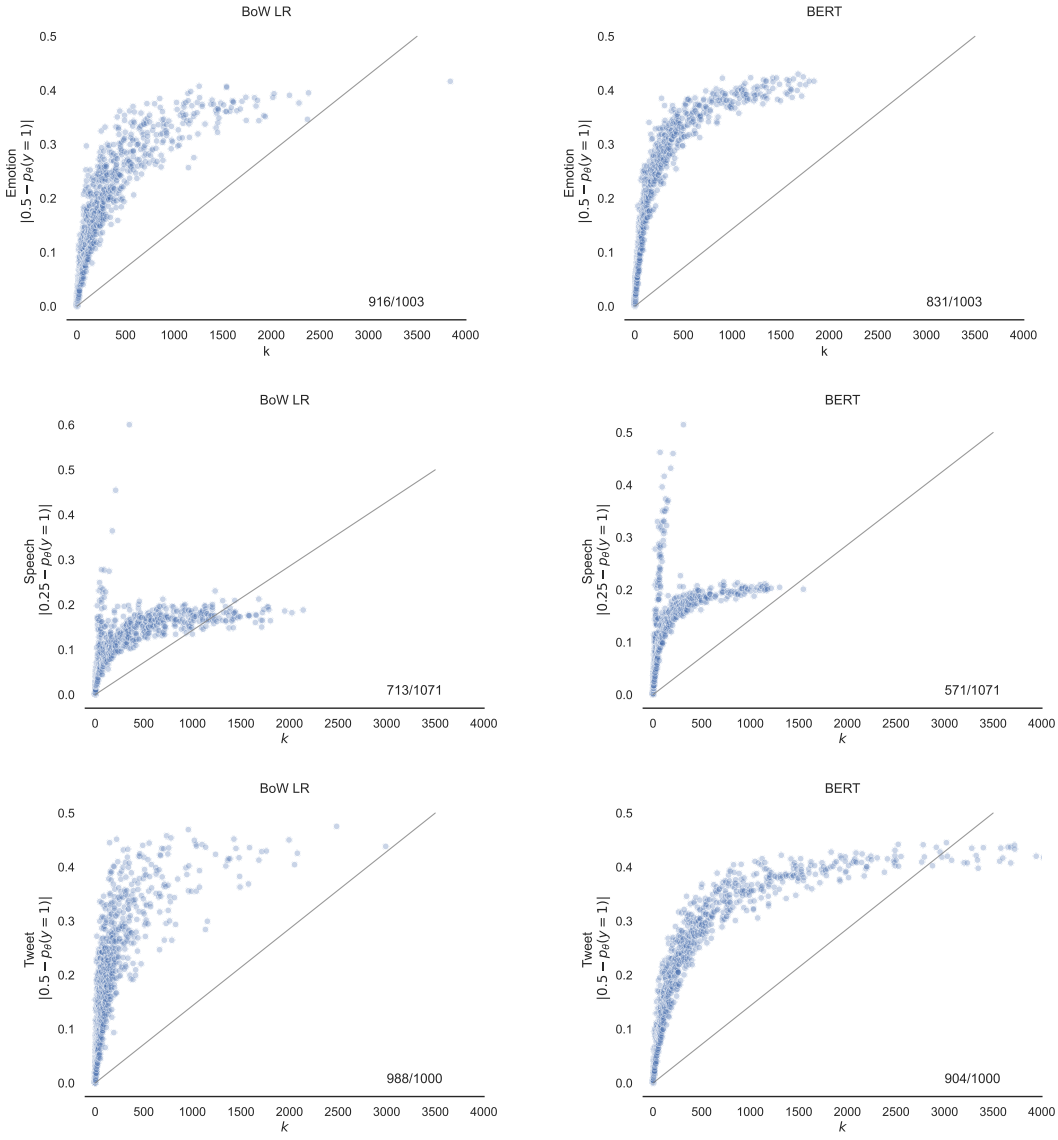


Figure A.5: Relationship between predicted probabilities and $k = |\mathcal{S}_t|$ identified from Algorithm 2.