EXPLORING THE LIME ALGORITHM FOR INTERPRETABLE DEEP LEARNING ON TEXT AND IMAGES

Anna Katharina Granberg Mortensen, Filip Sawicki, Javier García Ciudad, Thomas Theis Petersen

Master Students at the Technical University of Denmark

ABSTRACT

LIME (Local interpretable model-agnostic explanations) is an algorithm aimed at explaining predictions made by black-box models using other simpler interpretable models. In this paper it will be explored how several parameters of the algorithm influence explanations for a neural network in both image and text data, as well as, looking at how robust the algorithm is to these changes. These changes include testing Ridge Regression, Lasso Regression, Support Vector Regression and Decision Trees as interpretable models, including feature selection methods for the input given to these and also using different neighborhood generation techniques. Furthermore, some simpler experiments with different numbers of neighbors and the presence of weights were conducted. As a result, some of these changes have been found to be very influential on the explanations produced by LIME, while others are less so.

***Index Terms*—** LIME, explainability, deep learning, image classification, text classification

1. INTRODUCTION

The aim of LIME [1] is, given a prediction made by a black-box model, to train a local interpretable model which can be used to explain the prediction. The explanation is meant to be easily understandable by both machine learning experts and layman persons for proper evaluation of model's inference, and to inspire trust in the predictions. This idea can be used on any kind of data (tabular, text or images), though the implementation will be somewhat dependent on the data type, thus in this paper the focus is restricted to text and image data. In the baseline implementation of LIME there are several parameters that are open for the user to tune. But since, in most cases, there is no ground truth to be used for tuning these parameters, it would be desirable if the explanations provided by LIME were not too dependent on these choices [2]. Thus this paper seeks to investigate how robust the explanations obtained by using LIME are to variations of some of these parameters. To do this investigations both text and image data has been investigated. The concrete text and image on which the results presented in this paper are based can be found in Appendix A.

Thanks to Nicki Skafte for supervision throughout this project.

2. ALGORITHM

The first step of the algorithm is to produce some interpretable representation of the instance on which the prediction is based. For text this can be a tokenized version of the text and for images this is typically a super pixel segmentation. Then by perturbing this representation, a set of samples related to the original instance is obtained (neighborhood), and for each sample a weight is calculated based on their distance to the original instance. Each of these samples are then transformed back to the original representation, and input into the black-box model for classification. Based on these predictions, feature selection is done on the neighborhood data, before using the samples and predictions in the simple model. The model is then fitted on the remaining samples in the interpretable representation using the weights to give more influence to samples similar to the original instance. Here the probabilities made by the black-box model are used as targets. In multi-class problems one simple model will be fit for each class. Finally the original prediction can be explained by interpreting the simple model. Pseudocode for LIME can be found in Algorithm 1 and our implementation in Python is published on GitHub [3].

Algorithm 1 LIME

Require: x (instance to be explained), N (number of samples to use), k (number of elements to output), different functions and models as described in above.
  for $i \in \{1, 2, 3, ...N\}$ **do**
    $x'_i \leftarrow sample\_around(x_i)$
    $\pi_i \leftarrow distance(x_i, x'_i)$
    $t'_i \leftarrow predict(x'_i)$
  end for
  $\hat{x} \leftarrow feature\_selection(x')$
  $interpretable\_model.fit(\hat{x}, t', \pi)$
  $w \leftarrow interpretable\_model.coeficients$
  return $w.max(k)$

3. EVALUATION METRICS

3.1. Coefficient of Variation (CoV)

This coefficient is used as a measure of how relevant a LIME parameter is as well as how robust LIME is to changes in the parameter. It expresses how different the coefficients of the simple model are when changing a given parameter: If the CoV is large, it means the difference is large when changing a given parameter, and thus this parameter is very relevant for tuning the LIME algorithm. It is calculated as $CoV = \bar{\sigma}/\bar{\beta}$, where $\bar{\sigma}$ is the mean of the standard deviation of each coefficient and $\bar{\beta}$ is the mean of all the coefficients.

3.2. Intersection over Union (IoU)

This metric expresses the goodness of an explanation produced by LIME. The larger it is, the better the explanation is. It is calculated as $IoU = I/U$, where I is the intersection of the features considered important by LIME and the features considered important by a manually done annotation of the data, and U is the union of them.

4. EXPERIMENTS

Two different versions of LIME has been implemented in order to do the following three sets of experiments. The black-box model used of image classification was InceptionV3 which was pretrained on ImageNet dataset. For text classification a neural network with an embedding bag and one fully connected layer trained on the AGNews dataset was used.

4.1. Initial experiments

The initial experiments performed on the two LIME implementations were related to the results of changing the number of samples in the neighborhood generation and the effect of weighting the samples according to their 'closeness' to the initial instance. What was found was that the number of samples is an important parameter, where more samples lead to better and more robust explanations and that when the number of samples is high, weighting the samples makes little difference in the explanations. Further documentation of these experiments can be found in Appendix B.

4.2. Simple Explainer Model

4.2.1. Feature Selection

Due to the curse of dimensionality and computational costs of training simple models it is preferred to limit the number of input features given to the simple explainer models. Thus before proceeding with experimentation on interpretable models an experiment regarding feature selection was performed.

For the text based LIME algorithm two different feature selection methods were tested: Forward selection and highest weights, both using ridge regression with alpha penalty of: $0, 0.01, 0.1, 1, 10$, and random_uniform neighborhood generation. This test was not performed on the image data due to time complexity constraints. Overall no significant differences in word coefficients, CoV or IoU were found. However, the largest difference was time complexity. Forward selection was measured to be around 20 times slower than highest weights, which made it not worthy to be used in practice.

4.2.2. Simple Regression Model

For the experiments on the simple interpretable models, four different regression methods with varying hyperparameters were tested on text and image data. A qualitative inspection was done on the explanations to determine the best hyperparameters for each model.

Ridge: alpha $\in \{0.01, 0.1, 1, 10\}$
Using ridge regression as the simple model selected more features to be important compared to the three other methods and was quite robust as long as parameters were not extreme. Increasing the value of the alpha penalty made the coefficients slightly tend to 0. Based on the qualitative inspection it was found that alpha = 0.01 performed best for text based classification and alpha = 10 for image. Figure 1 shows a visual representation of the explanation produced by LIME on the Sci/Tec text data using the best ridge model.

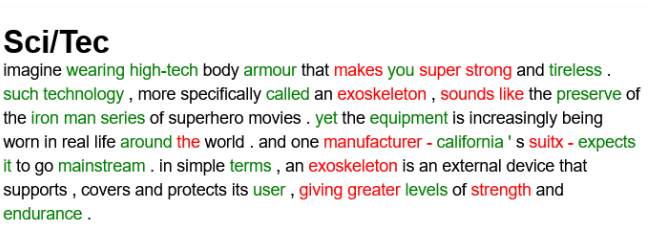


Fig. 1: Green words are words with a positive coefficient above 0.02, and red are words with coefficient less than -0.02

Lasso: alpha $\in \{0.001, 0.01, 0.1\}$
The results of lasso were similar to the ridge model, however higher values of alpha penalized coefficients more heavily and only the most important words and superpixels were selected. Using an alpha penalty above 0.01 resulted in all coefficient being exactly zero for both image and text data. Based on the qualitative inspection it was found that alpha = 0.001 performed best for both text and image based classification.

Linear-SVR: C $\in \{0.1, 1, 10, 100\}$
Linear-SVR had slightly different results compared to the ridge model. It was found to be sensitive to changes of C, and it was difficult to assess if the explanation was systematically meaningful or random. Based on the qualitative inspection it was found that C = 1 performed best for the text and classifi-

cation, while C = 100 was best for image based classification. It would have been interesting to try other kernels, however, this was not possible since the LIME algorithm requires the output coefficients to have the same dimension as input features.

Decision Tree [4]: *MD $\in \{3, 5, \text{None}\}$, **SS $\in \{2, 5\}$ *MD is maximum depth and "None" means that nodes are expanded until leaves are pure or contain less than SS samples, **SS is minimum sample split. The results obtained using decision tree regression were most different compared to the other methods. As tree based methods do not return coefficients but feature importances measured as the reduction in the criterion used to select split points (here using the Gini index), the representation of the explanations showed only most positively influencing words and superpixels. Moreover, for text based LIME the explanations seemed to catch important words that the other methods did not consider as important. Based on the qualitative inspection it was found that MD = None and SS = 2 performed best for both text and image based classification. Figure 2 shows a visual representation of the explanation produced by LIME on the Sci/Tec text data using the best decision tree model.

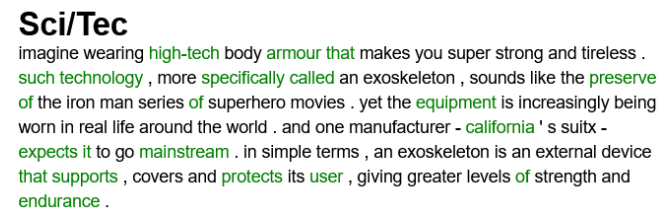


Fig. 2: Green words are words with a positive feature importance above 0.02

In Appendix C more visualizations of the outputs of the simple models with the best parameters can be seen for both text and image.

Having subjectively evaluated the difference in the models, and found the best parameters, the IoU for these four models when compared to the annotations were calculated. The results are presented in Table 1. Here it can be seen that for both text and image data the tree model performs best. However, since only few words/pixels were selected in the annotations, the IoU favors more selective explanations. And thus when subjectively comparing the explanations, ridge is actually found to also be reasonable on both text and images, even better than Lasso and SVR.

	Ridge	Lasso	SVR	Decision Tree
IoU Text	0.14	0.18	0.19	0.24
IoU Image	0.26	0.31	0.19	0.35

Table 1: Intersection over union done on best models vs annotations

Lastly looking at the CoV as presented in Table 2, it can be seen that LIME for text is somewhat robust against changes in the simple model, but that for images it is has a large impact. However, in the visual inspection of the text explanations, large differences were also found.

	Ridge vs Decision Tree	All Models (a)	All Models (b)
CoV Text	1.49	1.78	2.64
CoV Image	0.92	5.36	5.45

Table 2: Comparison of CoV of different explainer models. All Models (a) include only models with best parameters where (b) include all configurations discussed.

4.3. Neighborhood Generation

4.3.1. Text data

In the text version of LIME the neighborhood generation is done by creating samples where only certain words/tokens of the instance are active. Five different methods for neighborhood generation in text data has been investigated in this part, the different versions were: **(random_uniform)** which was based on uniformly random choosing if a token should be active in a given sample with a probability 0.5. This was the default option. **(random_normal)** randomly choosing a 'center token' and then using a normal distribution centered on that token, with a variable spread to choose which tokens should be active. **(one_on)** just one token active in each sample. **(one_off)** one token inactive in each sample. **(consecutive)** having a variable number of consecutive tokens active in each sample, systematically cycling all tokens. Note in the three last options, the number of samples would be reduced to equal the number of tokens, in the case where the number of tokens is smaller than the number of samples.

These options were tested on 1024 samples, with no weighting of the samples, using the 'highest coefficients' feature selection and using both ridge regression and and decision tree as the simple model. The spread for option two was chosen to be: $5, 10, 10$ and the parameter controlling how many consecutive tokens should be active in option five was chosen to be: $3, 10, 20, 40, 75$. In Table 3 it can be seen random_uniform(ru) and random_normal(rn) with a spread of 20 perform relatively well for both simple models, along with one_on and one_off for the decision tree.

Text	ru	one_on	one_off	rn-20	con-3
IoU Ridge	0.24	0.18	0.13	0.24	0.19
IoU Tree	0.25	0.21	0.22	0.22	0.18

Table 3: Intersection over union done on selected configurations vs annotation. Full data can be found in Appendix D

However, it is again observed that the IoU favors more selective explanations. When investigating which words are actually presented as most important, the best methods seem

to be: Random_uniform for both simple models, one_off for both simple models and random_normal with a spread of 20 for ridge regression. Random normal with a spread of 20 for the decision tree chooses some 'good' words, but is also give importance to some punctuation, which might not be so usable in an explanation presented to a person. Some examples of the visual representations on which this is based, can be found in Appendix D.

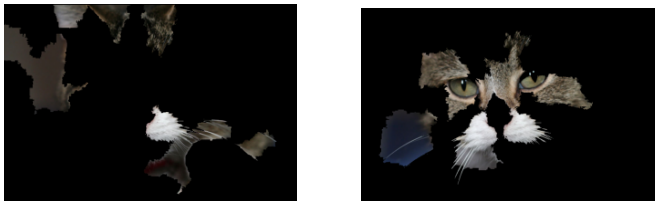
4.3.2. Image data

For image, the neighborhood generation is done by deactivating some superpixels. For this purpose, 4 different neighborhood generation methods were implemented. The one taken as the baseline is the same as in the text data: **(Random)** randomly deactivate each pixel with a probability of 0.5. The second method **(OneOn)** was to randomly choose one superpixel and leave it as the only active superpixel, while the third method **(OneOff)** works the other way around: only deactivate one superpixel, chosen randomly. The fourth method **(Radio)** was to randomly choose a superpixel, and activate not only that one but also those superpixels that were inside a given radio from the initial superpixel. In this last method, the distance between superpixel was calculated using the centroid of each superpixel. All four methods were tested using ridge regression and 50 neighbors, and a radio of 150 pixels in the case of that method.

The performance of each of these methods can be seen in Table 4 where the IoU of each is presented. As can be observed, both the random and radio methods have good performance, while the other two performed poorly. These results make sense since in the given segmentation the superpixels were small and the relevant structures were split among several superpixels (e.g. the whiskers were divided in 3 superpixels), thus, altering a superpixel is not very meaningful unless we also alter more superpixels around, which is not the case in OneOn and OneOff.

	Random	OneOn	OneOff	Radio
IoU Image	0.38	0.16	0.11	0.32

Table 4: Intersection over union for each image neighborhood generation type vs annotation



(a) OneOff (b) Radio

Fig. 3: Image explanations using two different neighborhood generation types. Original image is in Figure 4.

The difference in the explanations produced by two of the different generation methods is shown in Figure 3.

4.3.3. Coefficient of Variance

The coefficient of variance when changing the neighborhood generation method but keeping the simple model as the ridge regression and decision tree respectively, can be seen in Table 5. From this the conclusion would be that it is more important to choose the right neighborhood generation method when using ridge regression than when using a decision tree, but the difference between feature importance metric from the decision tree (which is only positive) and the coefficients from the ridge regression (which can be both positive or negative) distorts this comparison.

	Ridge	Decision Tree
CoV Text	46.75	1.29
CoV Image	7.27	1.18

Table 5: CoV for all different neighborhood generation methods for Ridge Regression and Decision Tree model

5. DISCUSSION AND CONCLUSION

The largest challenge in this paper has been to do a quantitative analysis of the explanations. For this the IoU was used but this lead to discussions about how the annotations should be made; should they be objective (matching the true underlying connections in the data, e.g. that '-' is used more often in science than other news genres), or should they match what a person would expect to be reasons for predictions. An idea for an alternative option, was to rank/weight words/pixels according to how much a person would expect them to be related to the prediction. Another limitation of the paper is the fact that just 2 images and 2 texts were primarily used throughout this exploration of the LIME algorithm, this could cause the results to be over fitted to the used data, and thus would not be representative. And lastly it was observed that the explanations were not entirely consistent between runs due to randomness introduced both in the segmentation of the images and in the neighborhood generation methods, to mitigated this more iterations should have been performed, and the mean calculated from that. That being said, for the data instances used here and based on the CoV, it has been found that the most important parameters to focus on are the simple model and the neighborhood generation method. For both the images and text data, the model with best performance based on IoU was the Decision Tree but from a human visual perspective the ridge seemed better. Regarding the neighborhood generation methods it was found that randomly choosing which words/superpixels should be active in each sample works really well, but that other methods should also be coincided, though these will be dependent on the given data type.

6. REFERENCES

[1] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," *CoRR*, vol. abs/1602.04938, 2016.

[2] Christoph Molnar, "Local surrogate (lime)," https://christophm.github.io/interpretable-ml-book/lime.html#, 2021.

[3] Katharina Granberg, Filip Sawicki, Javier García Ciudad, and Thomas Petersen, "02460-advanced-machine-learning," https://github.com/fpsawicki/02460-Advanced-Machine-Learning, 2021.

[4] Kacper Sokol and Peter Flach, "Limetree: Interactively customisable explanations based on local surrogate multi-output regression trees," *arXiv preprint arXiv:2005.01427*, 2020.

APPENDICES

A. DATA USED FOR REPORT PLOTS

A.1. News article for text examples

News article excerpt taken from BBC:
Imagine wearing high-tech body armour that makes you super strong and tireless. Such technology, more specifically called an exoskeleton, sounds like the preserve of the Iron Man series of superhero movies. Yet the equipment is increasingly being worn in real life around the world. And one manufacturer - California's SuitX - expects it to go mainstream. In simple terms, an exoskeleton is an external device that supports, covers and protects its user, giving greater levels of strength and endurance.

Annotation of words expected to contribute positively towards prediction of Science/Technology genre:
high-tech, strong, technology equipment, real, life, manufacturer, expects, external, device, protects, user, levels, strength.

A.2. Cat images used for examples

Fig. 4: Cat image used for experimentation

Fig. 5: Cat image used for experimentation

Fig. 6: Annotation mask to evaluate explanations on Figure 4

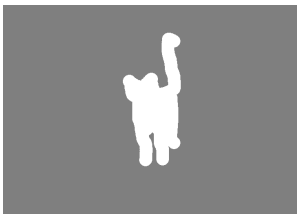


Fig. 7: Annotation mask to evaluate explanations on Figure 5

B. INITIAL EXPERIMENTS

B.1. Number of samples

The first experiment done on the text version of the LIME implementation was on how many samples generated by LIME were needed to get acceptable explanations from the simple models. The following number of samples were tested: $16, 32, 64, 128.256, 512$, while keeping all other parameters as their default setting. It was clear that with 16 and 32 samples the simple models were able to identify some of the relevant words, but too few to be usable. Increasing the number of samples made the simple models able to identify more important words and getting acceptable results. As can be seen from Figure 8, when increasing the number of samples the set of words identified as being positive contributors to the prediction, seemed to converge.

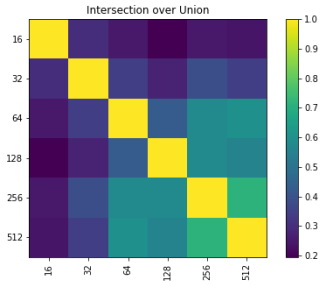


Fig. 8: Intersection over Union on pairs of configurations - text

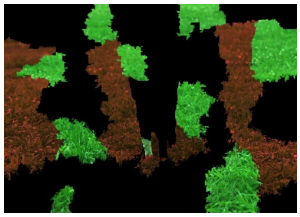
For the image version, this number was limited to around 70 samples due to RAM constraints, so 8 equally spaced values between 1 and 60 were tested. Although the number is much lower than in the text version, it was enough to get rather good explanations. The main finding was that the larger the number of samples, the better performance was achieved. This can be seen in Table 6. In Figure 9, it can be observed

Neigh.	1	9	17	26	34	43	51	60
IoU Image	0.13	0.11	0.16	0.25	0.26	0.26	0.26	0.29

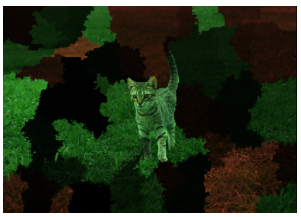
Table 6: Intersection over Union on different number of samples - images

how LIME assigns importance (bright green color) to pixels that are in the background when given just 1 neighbor. Then, as the number of samples increases, the algorithm progressively focuses on the cat and discards the superpixels in the background.

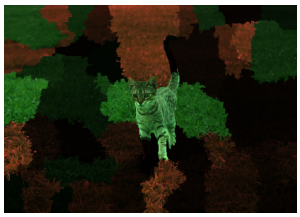
The CoV related to changes in the number of neighbors was found to be 10.07 for text and 3.05 for image, which are large values if compared to the CoV of other LIME parameters, thus it can be said that this is an influencial parameter.



(a) 1 neighbor



(b) 17 neighbors



(c) 43 neighbors

(d) 60 neighbors

Fig. 9: Image explanations with different number of neighbors. Green superpixels represent positive coefficients for the image being classified as a cat, and the red ones represent negative coefficients. At the same time, the brightness of the pixel expresses the magnitude of the coefficient.

B.2. Weighting Samples

For the second experiment the focus was on the influence of weighting the samples according to their 'closeness' to the instance.

For the text version the experiment was carried out by running an experiment similar to the one above, but testing the explanation obtained by each sample size both with the weighting of the samples turned on and off. For the larger sample sizes the affect of the weighting the samples was minimal. For the small sample sizes, 16 and 32, it was noticeable that weighting the samples made the simple models more selective when identifying important words, however, it was difficult to assess weather the weighting of the data made the simple models choose 'better' words.

For the image version, this was only carried out for the default number of samples (50), and it was found that the weights do not have a large influence on the result.

Specifically, for images the CoV related to using weights or not, was 1.24, which is a small value when compared to other LIME parameters. The CoV's for the text experiments can be seen in Table 7. Also here the value is quite small when the sample size is high, matching what was stated above.

Samples	16	32	64	128	256	512
CoV Text	6.68	7.18	6.79	3.07	1.34	1.17

Table 7: CoV related to weighting samples or not for LIME text

C. SIMPLE MODELS APPENDIX

Sci/Tec

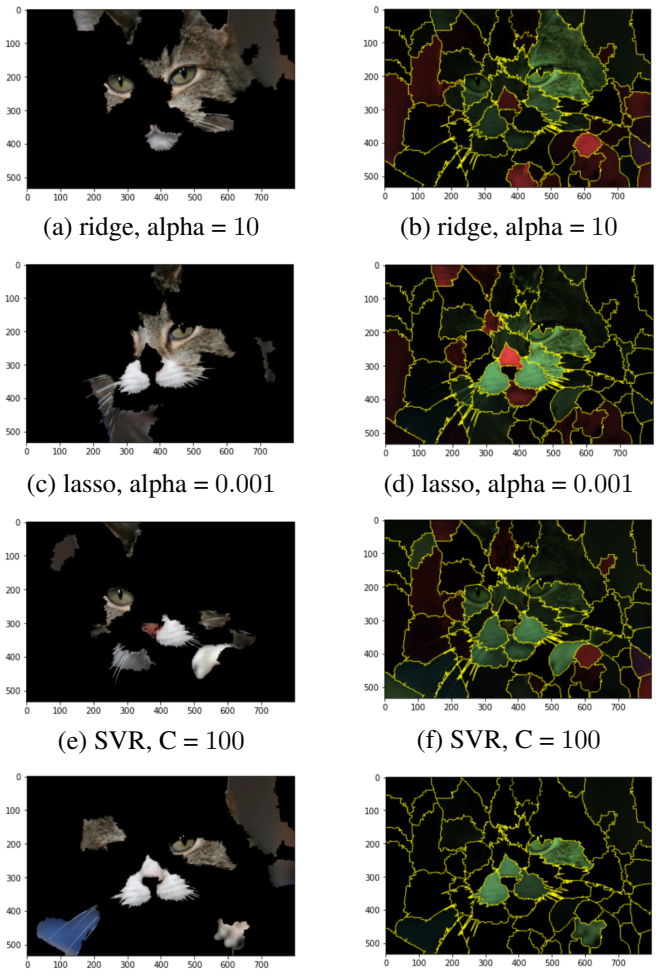
imagine wearing high-tech body armour that makes you super strong and tireless . such technology , more specifically called an exoskeleton , sounds like the preserve of the iron man series of superhero movies . yet the equipment is increasingly being worn in real life around the world . and one manufacturer - california ' s suitx - expects it to go mainstream . in simple terms , an exoskeleton is an external device that supports , covers and protects its user , giving greater levels of strength and endurance .

Fig. 10: Explanation made with lasso and alpha = 0.001. Green words are words with a positive coefficient above 0.02, and red are words with coefficient less than -0.02

Sci/Tec

imagine wearing high-tech body armour that makes you super strong and tireless . such technology , more specifically called an exoskeleton , sounds like the preserve of the iron man series of superhero movies . yet the equipment is increasingly being worn in real life around the world . and one manufacturer - california ' s suitx - expects it to go mainstream . in simple terms , an exoskeleton is an external device that supports , covers and protects its user , giving greater levels of strength and endurance .

Fig. 11: Explanation made with SVR and C = 100. Green words are words with a positive coefficient above 0.02, and red are words with coefficient less than -0.02



(a) ridge, alpha = 10 (b) ridge, alpha = 10

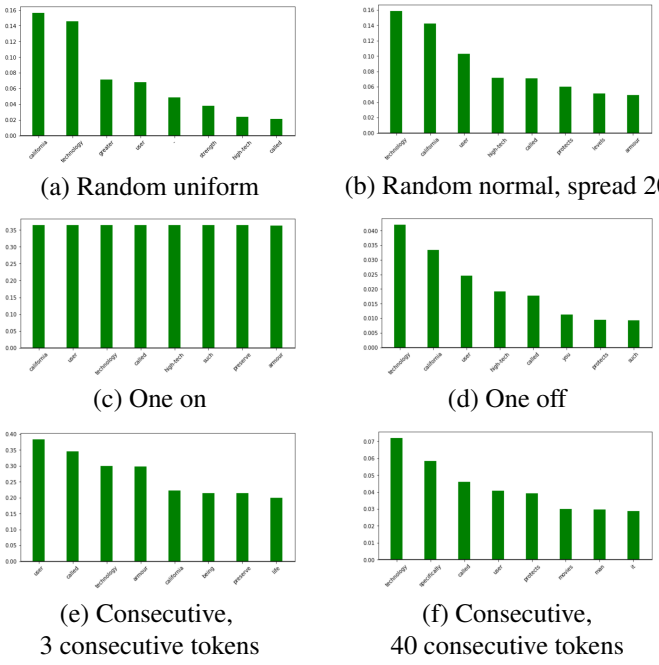
(c) lasso, alpha = 0.001 (d) lasso, alpha = 0.001

(e) SVR, C = 100 (f) SVR, C = 100

(g) decision tree, MD = None, SS = 2 (h) decision tree, MD = None, SS = 2

Fig. 12: Visual representation of different simple models used in LIME in image data with the best parameters

D. NEIGHBORHOOD GENERATION APPENDIX

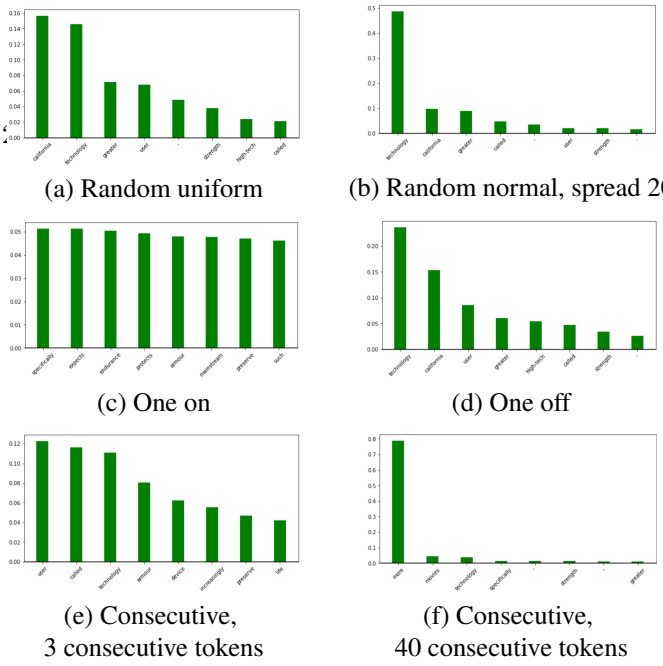


(a) Random uniform (b) Random normal, spread 20

(c) One on (d) One off

(e) Consecutive, 3 consecutive tokens (f) Consecutive, 40 consecutive tokens

Fig. 13: Eight words with highest coefficients for different neighborhood generation methods - ridge regression



(a) Random uniform (b) Random normal, spread 20

(c) One on (d) One off

(e) Consecutive, 3 consecutive tokens (f) Consecutive, 40 consecutive tokens

Fig. 14: Eight words with highest feature importance for different neighborhood generation methods - decision tree

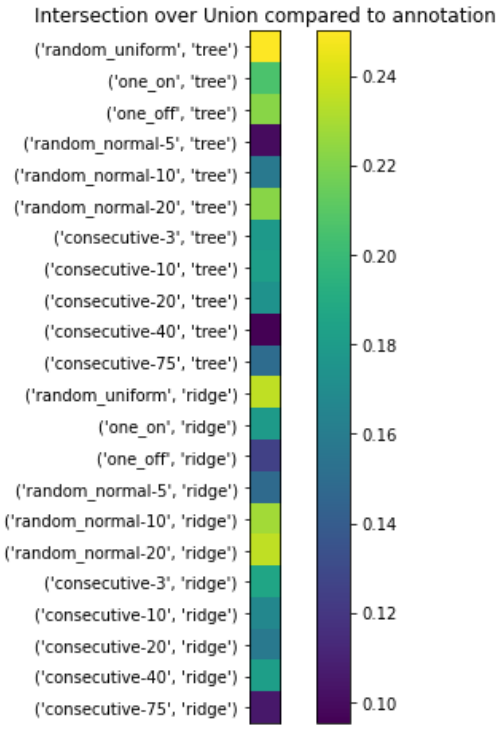


Fig. 15: Intersection over union done on each configuration of neighborhood generation model and simple model vs annotation on text data