Identifying Semantically Difficult Samples to Improve Text Classification

Anonymous ACL submission

Abstract

In this paper, we investigate the effect of addressing difficult samples from a given text dataset on the downstream text classification task. We define difficult samples as being non-obvious cases for text classification by analysing them in the semantic embedding space; specifically - (i) semantically similar samples that belong to different classes and (ii) semantically dissimilar samples that belong to the same class. We propose a penalty function to measure the overall difficulty score of every sample in the dataset. We conduct exhaustive experiments on 13 standard datasets to show a consistent improvement of up to 9% and discuss qualitative results to show effectiveness of our approach in identifying difficult samples for a text classification model.

1 Introduction

In the recent past there has been an emphasis on the assessment of quality of data for machine learning tasks (Jain et al., 2020)(Swayamdipta et al., 2020) and a few approaches focus on assessing the training datasets. (Ghorbani and Zou, 2019),(Yoon et al., 2020) have looked at the problem of finding most valuable data instances for a chosen classifier. (Csáky et al., 2019) discuss a method for data filtering to improve the quality of data for neural conversation models in a model agnostic fashion. (Peinelt et al., 2019) suggest profiling the datasets to find non obvious cases for semantic similarity datasets. In this paper, we present our analysis of semantically difficult samples in the training data and their impact on the downstream models for text classification task. Table 1 shows examples of two types of difficult samples - (i) samples with high semantic similarity and different labels and (ii) samples with low semantic similarity and belonging to the same class. We propose an intuitive penalty function to measure the difficulty score of every sample in the dataset. We present both quantitative and qualitative results to study the effect of samples from both these categories on the performance of the downstream text classifiers. Similar to (Csáky et al., 2019), our method is model agnostic. We present our results on 13 standard datasets utilizing standard text classifiers and encoding schemes.

Class	*Original Sample*
`Neutral`	@AmericanAir Thank you
`Positive`	@USAirways Thank you
`Neutral`	@united what's a girl gotta do to get a flight name change when SHE bought one for a mean ex boyfriend and needs a girl's trip stat?!
`Neutral`	@AmericanAir @pbpinftworth iPhone 6 64GB (not 6 plus)

Table 1: Samples from Airline Tweets Dataset

2 Proposed Approach

2.1 Difficulty of a Sample

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ be a labelled train dataset with n samples where x_i being the input text example and y_i its corresponding label. Let e_i denote the encoded vector representation of the input text x_i. For a pair of samples in the embedding space (e_i, e_j), we argue for the following two cases that contribute to their difficulty score,

1. e_i and e_j are semantically similar (lie close to each other in the embedding space) but they have different labels ($y_i \neq y_j$) **[case 1]**
2. e_i and e_j are semantically dissimilar (lie far apart in the embedding space) but they have the same label ($y_i = y_j$) **[case 2]**

It is intuitive as to why samples belonging to *case 1* can be difficult. For *case 2*, while one can expect semantic dissimilarity between a class, we look at the extreme cases, where the samples even though are part of the same class, could be referring to

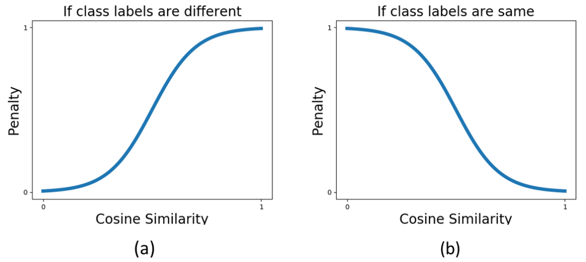

Figure 1: (a) S-penalty function (b) Z-penalty function

Algorithm 1: Identify Difficult Samples

Input : Labelled Text Dataset $\mathcal{D}$
Output : Difficult Samples
$[\mathcal{E}] \leftarrow$ Text embeddings for each sample.
for $i = 1, 2, \ldots |\mathcal{E}|$ **do**
  $cp_i = 0$
  for $j = 1, 2, \ldots |\mathcal{E}|, j \neq i$ **do**
    $x = cos(e_i, e_j) = \frac{e_i \cdot e_j}{||e_i|| \cdot ||e_j||}$
    Compute Pairwise Penalty
    if $y_i \neq y_j$, y_i is label for e_i **then**
      $S(x) = \frac{1}{1+e^{(a-bx)}}$
      cp_i += $S(x)$
    else if $y_i == y_j$ **then**
      $Z(x) = \frac{1}{1+e^{-(a-bx)}}$
      cp_i += $Z(x)$
    end if
  end for
  Store cumulative penalty for each sample
  $CP[i] = cp_i$
end for
Sort CP in descending order
Return top $k\%$ samples from $\mathcal{D}$ using penalty scores from CP

two different concepts within the same class. For a given sample e_i, we consider its pairwise relationship (PR) with all other samples in the dataset to compute its overall difficulty score. However, the intention is to only penalise the sample pair if it meets the above criterion (case 1&2). Thus, we introduce penalty functions (see Figure 1) that consider the PR and corresponding labels to output a penalty score for the input pair (e_i, e_j).

2.2 Penalty Function

For a sample pair (e_i, e_j), the PR is captured by the cosine similarity $cos(e_i, e_j) = \frac{e_i \cdot e_j}{||e_i|| ||e_j||}$. Instead of determining a threshold on the cos_{sim} to identify if the samples are similar or dissimilar, we employ a sigmoid function to assign the penalty scores. Specifically, we use an *s-shaped* sigmoid $S(x) = \frac{1}{1+e^{(a-bx)}}$ for case 1 and a *z-shaped* sigmoid $Z(x) = \frac{1}{1+e^{-(a-bx)}}$ for case 2 where $a, b > 0$ and x is the cos_{sim}. Thus, if the samples belong to different classes but have a high cos_{sim}, utilizing $S(x)$, a high penalty value is assigned and vice-versa. Similarly, if the samples belong to the same class but have high cos_{sim}, utilizing $Z(x)$, a low penalty value is assigned and vice-versa.

2.3 Identifying Difficult Samples

For each sample in the dataset, a cumulative penalty score is computed by summing the pairwise penalty scores with all the other samples in the dataset. The samples in the dataset are sorted in descending order w.r.t. the cumulative penalty scores and the top $k\%$ samples are labelled as difficult. The overall approach is summarized in Algorithm 1.

3 Datasets and Experiments

3.1 Datasets

We identify 13 standard datasets used for text classification from prior art literature as shown in Table 2 and use the standard split for train, validation, and test sets when available. For AT (Rane and Kumar, 2018), MR (Pang et al., 2002a) and SE (Nakov et al., 2019) a 10% split from training set has been used as a validation set.

3.2 Experiment Setup

3.2.1 Data Preprocessing

For all the experiments, standard pre-processing steps such as removal of special characters, stop-words, conversion to lowercase, tokenization, etc. have been performed. For generating text representation, we consider two encoding strategies - (i) Average pre-trained word embeddings for tokens in input text using Word2Vec (Mikolov et al., 2013) and (ii) intermediate layer representation of trained Long Short Term Memory (LSTM) model (Hochreiter and Schmidhuber, 1997). For computing penalty scores using sigmoid functions we use $a = 5$ and $b = 10$.

3.2.2 Evaluation Strategy

For each dataset, we identify the top $k\%$ difficult samples as described in Sec 2.3 from the training set for $k \in [0, 1, 3, 5, 10, 20]$. We train two sets of classifiers, one with the complete training set and second after filtering the training set of

$Dataset$	$Source$	$Classes$	$Size$	$F1_{0\%}$	$F1_{1\%}$	$F1_{5\%}$	$F1_{Best}$ $(k\%)$
AT	Rane and Kumar	3	14640	71.76%	71.89%	**72.44%**	**72.44% (5%)**
CM	Collins et al.	4	5218	**93.73%**	92.92%	87.04%	**93.73% (0%)**
CB	Uzzi et al.	4	32000	99.98%	**99.99%**	**99.99%**	**99.99% (1%)**
HS	Davidson et al.	3	20941	69.06%	68.65%	**69.95%**	**69.95% (5%)**
MR	Pang et al.	2	2000	72.61%	75.41%	76.22%	**81.74% (3%)**
POL	Pang et al.	2	1400	70.70%	**71.56%**	70.5%	**71.56% (1%)**
PSC	Collins et al.	5	3117	59.05%	59.18%	61.63%	**68.17% (3%)**
QC	Li and Roth	6	5142	86.88%	86.51%	**87.00%**	**87.00% (5%)**
RS	Kotzias et al.	2	3000	79.82%	**80.82%**	78.64%	**80.82% (1%)**
SE	Nakov et al.	3	13231	57.68%	56.56%	58.85%	**58.95% (10%)**
SMSS	Almeida et al.	2	9416	**97.78%**	**97.78%**	96.81%	**97.78% (1%)**
YTS	Alberto et al.	2	1948	94.78%	**96.64%**	95.11%	**96.64% (1%)**
20NG	Adi and Çelebi	20	20000	59.11%	57.59%	59.21%	**60.26% (3%)**

Table 2: Performance comparison of trained LSTM models when top $k\%$ difficult samples are removed from the training set. $F1_{Best}$ represents the best score observed for $k \in [0, 1, 3, 5, 10, 20]$. $F1_{0\%}$ represents baseline performance when no samples are removed.

the difficult samples using different values of k and compare the macro F1 scores for both sets of classifiers on held out test sets. For text representations generated using Word2Vec, we utilize an SVM model with RBF kernel with hyperparameters C & $\gamma \in [0.001, 0.01, 0.1, 1, 10, 100]$ that are tuned using grid search on the validation set. The LSTM model is trained with the embedding layer initialized with one-hot vector representation of the input text where the maximum vocabulary size is $V_{max} = 10000$ and the maximum sequence length is $S_{max} = 250$ and the network parameters are tuned on the validation set.

4 Results and Discussion

4.1 Analysis of Removing Difficult Samples

As seen from the results shown in Table 2, an improvement in the F1 scores (upto ∼9%) is seen for most of the datasets. The values for $F1_{Best}$ indicate that the improvement is generally observed for $k \in [1, 3, 5]$. Although we saw a consistent improvement in the performance of the trained models on the updated train set (difficult samples removed) for lower values of k, we observed a consistent drop in performance when top 10% or 20% difficult samples were removed. This suggests that the top-ranked difficult samples in the training set ($k \in [1, 3, 5]$) help in model generalization while for $k > 5$ the trained models seem to overfit on the training set resulting in poor performance on the test set. The CM dataset (Collins et al., 2018) is an exception to this general trend. On further investigation, we observed that the top ranked difficult samples belong to a minority class and removing them hurts the model performance for test samples from that class. Thus, for text datasets, we observe that semantically difficult samples pose a challenge for the downstream model and removing them in most cases improves its performance.

4.2 Analysis of Difficult Samples

Figure 2 shows the various regions in the TSNE plots corresponding to the top-ranked difficult samples belonging to various cases as discussed in Sec 2.1. The difficult samples due to *case 1* (Figure 2(c)) belong to the region of maximum overlap between the class distributions. On the contrary, difficult samples due to *case 2* (Figure 2(d)) belong to isolated clusters of samples that are not representative of the respective overall class distributions. Figure 2(b) shows the region of difficult samples when both cases are considered for YTS dataset. Figure 3 shows the trend of the F1 scores for all three cases. As observed, the performance gain is higher when difficult samples from both cases are considered than difficult samples from only *case 1* or *case 2* for $k \in [1, 3, 5]$, while for $k > 5$, the performance dips for all configurations.

Table 3 showcases a few examples of top-ranked difficult samples identified from YTS dataset. As seen from examples of Spam class, the constituent words are common with the Not Spam class and the overall semantic meaning does not specifically indicate the label Spam. These particular samples

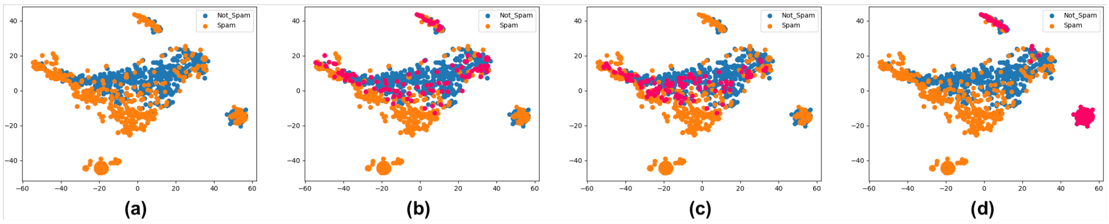

Figure 2: (a) Original TSNE plot for YTS (b) Both cases - top 10% difficult samples (c) Case-1 only - top 10% difficult samples (d) Case-2 only - top 10% difficult samples (Difficult samples are marked in **pink**)

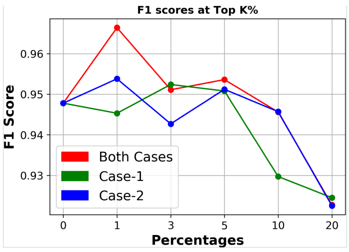

Figure 3: YTS Dataset - Plot of F1 scores for all cases for $k \in [0, 1, 3, 5, 10, 20]$. (Baseline accuracy $F1_{0\%}$ = 0.9478 for all cases).

$Dataset$	$F1_{0\%}^{W2V}$	$F1_{Best}^{W2V}$
`RS`	53.33%	58.50% (1%)
`SMSS`	94.78%	95.95% (1%)
`YTS`	84.91%	86.69% (1%)

Table 4: Performance of trained SVM model using Word2Vec embeddings.

lie in the overlap region as shown in Figure 2(b) and removing them from the training set improves the generalizability of the trained model. Similar insights can be derived for the difficult samples belonging to `Not Spam` class.

4.3 Effect of Text Encoding and Model

Table 4 shows the F1 scores obtained after removing difficult samples identified using Word2Vec embeddings and SVM classifier as discussed in Sec 3.2. We observe a similar improvement in the model performance after removing the top 1% difficult samples as shown in Table 2. We replicated the experiments to use TF-IDF (Sammut and Webb, 2010) as well as Fasttext (Bojanowski et al., 2017) embeddings and observed a similar trend. Thus, the identification of difficult samples is agnostic to the underlying text encoding scheme and improves the performance of general text classification model.

$Class$	$Original\ Sample$
`Spam`	He gets more views but has less subscribers lol
`Spam`	Yea stil the best WK song ever Thumbs up of you think the same ï»¿
`Not Spam`	Hello Brazil ðŸ˜»âœŒðŸ’“ðŸ˜
`Not Spam`	We get it, you came here for the views... Ôªø

Table 3: Top ranked difficult samples in YTS using our approach

4.4 Application to Human In Loop Systems

Difficult samples in a dataset could arise due to the data gathering process which might induce noisy labels. It can also be an intrinsic property of the dataset where gathered labels are not sufficient to capture the overall set of semantic topics and fine-grained labels are necessary. As seen from our analysis, the difficult samples are most likely to get misclassified by the trained model. Thus, our approach can quickly identify the semantically difficult samples in a dataset which a data scientist could use to review their labels with the help of a domain expert or build provisions in the modeling pipeline to address them.

5 Conclusion and Future work

We present a method to identify semantically difficult samples and suggest two scenarios of how such samples can affect the training data and the corresponding model. We show by extensive experimental evaluation that the classifiers trained after removing difficult samples show a gain in performance (∼9%) as compared to the classifiers trained on the full training set. Thus, we show that training data assessment is an important pre-step before training classifier models. A related problem is to automatically identify the optimum value of k for a dataset and remediate the data of the difficult samples, which we plan to explore in the future.

References

Abdulwahab O Adi and Erbuğ Çelebi. 2014. Classification of 20 news group with naïve bayes classifier. In *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, pages 2150–2153. IEEE.

Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. 2015. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pages 138–143. IEEE.

Tiago A Almeida, José María G Hidalgo, and Akebo Yamakami. 2011. Contributions to the study of sms spam filtering: new collection and results. In *Proceedings of the 11th ACM symposium on Document engineering*, pages 259–262.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information.

Edward Collins, Nikolai Rozanov, and Bingbing Zhang. 2018. Evolutionary data measures: Understanding the difficulty of text classification tasks. *arXiv preprint arXiv:1811.01910*.

Richárd Csáky, Patrik Purgai, and Gábor Recski. 2019. Improving neural conversational models with entropy-based data filtering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5650–5669.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning.

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

Abhinav Jain, Hima Patel, Lokesh Nagalapatti, Nitin Gupta, Sameep Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Ruhi Sharma Mittal, and Vitobha Munigala. 2020. Overview and importance of data quality for machine learning tasks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3561–3562.

Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 597–606.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2019. Semeval-2016 task 4: Sentiment analysis in twitter. *arXiv preprint arXiv:1912.01973*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002a. Thumbs up? sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002b. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Nicole Peinelt, Maria Liakata, and Dong Nguyen. 2019. Aiming beyond the obvious: Identifying non-obvious cases in semantic similarity datasets. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2792–2798.

Ankita Rane and Anand Kumar. 2018. Sentiment classification system of twitter data for us airline service analysis. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 769–773. IEEE.

Claude Sammut and Geoffrey I. Webb, editors. 2010. *TF–IDF*, pages 986–987. Springer US, Boston, MA.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

Brian Uzzi, Lise Getoor, Evimaria Terzi, and Lada Adamic. 2016. Four keynote speeches [2016 ieee/acm international conference on advances in social networks analysis and mining]. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages xl–xliii. IEEE Computer Society.

Jinsung Yoon, Sercan Arik, and Tomas Pfister. 2020. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, pages 10842–10851. PMLR.