# Evolutionary Data Measures: Understanding the Difficulty of Text Classification Tasks

**Edward Collins**
Wluper Ltd.
London, United Kingdom
ed@wluper.com

**Nikolai Rozanov**
Wluper Ltd.
London, United Kingdom
nikolai@wluper.com

**Bingbing Zhang**
Wluper Ltd.
London, United Kingdom
bingbing@wluper.com

## Abstract

Classification tasks are usually analysed and improved through new model architectures or hyperparameter optimisation but the underlying properties of datasets are discovered on an ad-hoc basis as errors occur. However, understanding the properties of the data is crucial in perfecting models. In this paper we analyse exactly which characteristics of a dataset best determine how difficult that dataset is for the task of text classification. We then propose an intuitive measure of difficulty for text classification datasets which is simple and fast to calculate. We show that this measure generalises to unseen data by comparing it to state-of-the-art datasets and results. This measure can be used to analyse the precise source of errors in a dataset and allows fast estimation of how difficult a dataset is to learn. We searched for this measure by training 12 classical and neural network based models on 78 real-world datasets, then use a genetic algorithm to discover the best measure of difficulty. Our difficulty-calculating code[1] and datasets[2] are publicly available.

## 1 Introduction

If a machine learning (ML) model is trained on a dataset then the same machine learning model on the same dataset but with more granular labels will frequently have lower performance scores than the original model (see results in Zhang et al. (2015); Socher et al. (2013a); Yogatama et al. (2017); Joulin et al. (2016); Xiao and Cho (2016); Conneau et al. (2017)). Adding more granularity to labels makes the dataset harder to learn - it increases the dataset's difficulty. It is obvious that some datasets are more difficult for learning models than others, but is it possible to quantify this difficulty? In order to do so, it would be necessary to understand exactly what characteristics of a dataset are good indicators of how well models will perform on it so that these could be combined into a single measure of difficulty.

Such a difficulty measure would be useful as an analysis tool and as a performance estimator. As an analysis tool, it would highlight precisely what is causing difficulty in a dataset, reducing the time practitioners need spend analysing their data. As a performance estimator, when practitioners approach new datasets they would be able to use this measure to predict how well models are likely to perform on the dataset.

The complexity of datasets for ML has been previously examined (Ho and Basu, 2002; Mansilla and Ho, 2004; Bernadó-Mansilla and Ho, 2005; Maci et al., 2008), but these works focused on analysing feature space data $\in \mathbb{R}^n$. These methods do not easily apply to natural language, because they would require the language be embedded into feature space in some way, for example with a word embedding model which introduces a dependency on the model used. We extend previous notions of difficulty to English language text classification, an important component of natural language processing (NLP) applicable to tasks such as sentiment analysis, news categorisation and automatic summarisation (Socher et al., 2013a; Antonellis et al., 2006; Collins et al., 2017). All of our recommended calculations depend only on counting the words in a dataset.

### 1.1 Related Work

One source of difficulty in a dataset is mislabelled items of data (noise). Brodley and Friedl (1999) showed that filtering noise could produce large gains in model performance, potentially yielding larger improvements than hyperparameter optimisation (Smith et al., 2014). We ignored noise in

[1] https://github.com/Wluper/edm
[2] http://data.wluper.com

this work because it can be reduced with proper data cleaning and is not a part of the true signal of the dataset. We identified four other areas of potential difficulty which we attempt to measure:

**Class Interference.** Text classification tasks to predict the 1 - 5 star rating of a review are more difficult than predicting whether a review is positive or negative (Zhang et al., 2015; Socher et al., 2013a; Yogatama et al., 2017; Joulin et al., 2016; Xiao and Cho, 2016; Conneau et al., 2017), as reviews given four stars share many features with those given five stars. Gupta et al. (2014) describe how as the number of classes in a dataset increases, so does the potential for ”confusability” where it becomes difficult to tell classes apart, therefore making a dataset more difficult. Previous work has mostly focused on this confusability - or class interference - as a source of difficulty in machine learning tasks (Bernadó-Mansilla and Ho, 2005; Ho and Basu, 2000, 2002; Elizondo et al., 2009; Mansilla and Ho, 2004), a common technique being to compute a minimum spanning tree on the data and count the number of edges which link different classes.

**Class Diversity.** Class diversity provides information about the composition of a dataset by measuring the relative abundances of different classes (Shannon, 2001). Intuitively, it gives a measure of how well a model could do on a dataset without examining any data items and always predicting the most abundant class. Datasets with a single overwhelming class are easy to achieve high accuracies on by always predicting the most abundant class. A measure of diversity is one feature used by Bingel and Søgaard (2017) to identify datasets which would benefit from multi-task learning.

**Class Balance.** Unbalanced classes are a known problem in machine learning (Chawla et al., 2004, 2002), particularly if classes are not easily separable (Japkowicz, 2000). Underrepresented classes are more difficult to learn because models are not exposed to them as often.

**Data Complexity.** Humans find some pieces of text more difficult to comprehend than others. How difficult a piece of text is to read can be calculated automatically using measures such as those proposed by Mc Laughlin (1969); Senter and Smith (1967); Kincaid et al. (1975). If a piece of text is more difficult for a human to read and understand, the same may be true for an ML model.

## 2 Method

We used 78 text classification datasets and trained 12 different ML algorithms on each of the datasets for a total of 936 models trained. The highest achieved macro F1 score (Powers, 2011), on the test set for each model was recorded. Macro F1 score is used because it is valid under imbalanced classes. We then calculated 48 different statistics which attempt to measure our four hypothesised areas of difficulty for each dataset. We then needed to discover which statistic or combination thereof correlated with model F1 scores.

We wanted the discovered difficulty measure to be useful as an analysis tool, so we enforced a restriction that the difficulty measure should be composed only by summation, without weighting the constituent statistics. This meant that each difficulty measure could be used as an analysis tool by examining its components and comparing them to the mean across all datasets.

Each difficulty measure was represented as a binary vector of length 48 - one bit for each statistic - each bit being 1 if that statistic was used in the difficulty measure. We therefore had $2^{48}$ possible different difficulty measures that may have correlated with model score and needed to search this space efficiently.

Genetic algorithms are biologically inspired search algorithms and are good at searching large spaces efficiently (Whitley, 1994). They maintain a population of candidate difficulty measures and combine them based on their ”fitness” - how well they correlate with model scores - so that each ”parent” can pass on pieces of information about the search space (Jiao and Wang, 2000). Using a genetic algorithm, we efficiently discovered which of the possible combinations of statistics correlated with model performance.

### 2.1 Datasets

We gathered 27 real-world text classification datasets from public sources, summarised in Table 1; full descriptions are in Appendix A.

We created 51 more datasets by taking two or more of the original 27 datasets and combining all of the data points from each into one dataset. The label for each data item was the name of the dataset which the text originally came from. We combined similar datasets in this way, for example

| Dataset Name | Num. Class. | Train Size | Valid Size | Test Size |
|---|---|---|---|---|
| AG's News (Zhang et al., 2015) | 4 | 108000 | 12000 | 7600 |
| Airline Twitter Sentiment (FigureEight, 2018) | 3 | 12444 | - | 2196 |
| ATIS (Price, 1990) | 26 | 9956 | - | 893 |
| Corporate Messaging (FigureEight, 2018) | 4 | 2650 | - | 468 |
| ClassicLit | 4 | 40489 | 5784 | 11569 |
| DBPedia (wiki.dbpedia.org, 2015) | 14 | 50400 | 5600 | 7000 |
| Deflategate (FigureEight, 2018) | 5 | 8250 | 1178 | 2358 |
| Disaster Tweets (FigureEight, 2018) | 2 | 7597 | 1085 | 2172 |
| Economic News Relevance (FigureEight, 2018) | 2 | 5593 | 799 | 1599 |
| Grammar and Product Reviews (Datafiniti, 2018) | 5 | 49730 | 7105 | 14209 |
| Hate Speech (Davidson et al., 2017) | 3 | 17348 | 2478 | 4957 |
| Large Movie Review Corpus (Maas et al., 2011) | 2 | 35000 | 5000 | 10000 |
| London Restaurant Reviews (TripAdvisor[3]) | 5 | 12056 | 1722 | 3445 |
| New Year's Tweets (FigureEight, 2018) | 10 | 3507 | 501 | 1003 |
| New Year's Tweets (FigureEight, 2018) | 115 | 3507 | 501 | 1003 |
| Paper Sent. Classification (archive.ics.uci.edu, 2018) | 5 | 2181 | 311 | 625 |
| Political Social Media (FigureEight, 2018) | 9 | 3500 | 500 | 1000 |
| Question Classification (Li and Roth, 2002) | 6 | 4906 | 546 | 500 |
| Review Sentiments (Kotzias et al., 2015) | 2 | 2100 | 300 | 600 |
| Self Driving Car Sentiment (FigureEight, 2018) | 6 | 6082 | - | 1074 |
| SMS Spam Collection (Almeida and Hidalgo, 2011) | 2 | 3901 | 558 | 1115 |
| SNIPS Intent Classification (Coucke, 2017) | 7 | 13784 | - | 700 |
| Stanford Sentiment Treebank (Socher et al., 2013a) | 3 | 236076 | 1100 | 2210 |
| Stanford Sentiment Treebank (Socher et al., 2013a) | 2 | 117220 | 872 | 1821 |
| Text Emotion (FigureEight, 2018) | 13 | 34000 | - | 6000 |
| Yelp Reviews (Yelp.com, 2018) | 5 | 29250 | 3250 | 2500 |
| YouTube Spam (Alberto et al., 2015) | 2 | 1363 | 194 | 391 |

Table 1: The 27 different publicly available datasets we gathered with references.

two different datasets of tweets, so that the classes would not be trivially distinguishable - there is no dataset to classify text as either a tweet or Shakespeare for example as this would be too easy for models. The full list of combined datasets is in Appendix A.2.

Our datasets focus on short text classification by limiting each data item to 100 words. We demonstrate that the difficulty measure we discover with this setup generalises to longer text classification in Section 3.1. All datasets were lowercase with no punctuation. For datasets with no validation set, 15% of the training set was randomly sampled as a validation set at runtime.

## 2.2 Dataset Statistics

We calculated 12 distinct statistics with different n-gram sizes to produce 48 statistics of each dataset. These statistics are designed to increase in value as difficulty increases. The 12 statistics are described here and a listing of the full 48 is in Appendix B in Table 5. We used n-gram sizes from unigrams up to 5-grams and recorded the average of each statistic over all n-gram sizes. All probability distributions were count-based - the probability of a particular n-gram / class / character was the count of occurrences of that particular entity divided by the total count of all entities.

### 2.2.1 Class Diversity

We recorded the Shannon Diversity Index and its normalised variant the Shannon Equitability (Shannon, 2001) using the count-based probability distribution of classes described above.

### 2.2.2 Class Balance

We propose a simple measure of class imbalance:

$$Imbal = \sum_{c=1}^{C} \left| \frac{1}{C} - \frac{n_c}{T_{DATA}} \right| \qquad (1)$$

$C$ is the total number of classes, $n_c$ is the count of items in class $c$ and $T_{DATA}$ is the total number of data points. This statistic is 0 if there are an equal number of data points in every class and the upper bound is $2\left(1 - \frac{1}{C}\right)$ and is achieved when one class has all the data points - a proof is given in Appendix B.2.

### 2.2.3 Class Interference

Per-class probability distributions were calculated by splitting the dataset into subsets based on the class of each data point and then computing count-based probability distributions as described above for each subset.

**Hellinger Similarity** One minus both the average and minimum Hellinger Distance (Le Cam and Yang, 2012) between each pair of classes. Hellinger Distance is 0 if two probability distributions are identical so we subtract this from 1 to give a higher score when two classes are similar giving the Hellinger Similarity. One minus the minimum Hellinger Distance is the maximum Hellinger Similarity between classes.

**Top N-Gram Interference** Average Jaccard similarity (Jaccard, 1912) between the set of the top 10 most frequent n-grams from each class. N-grams entirely composed of stopwords were ignored.

**Mutual Information** Average mutual information (Cover and Thomas, 2012) score between the set of the top 10 most frequent n-grams from each class. N-grams entirely composed of stopwords were ignored.

#### 2.2.4 Data Complexity

**Distinct n-grams : Total n-grams** Count of distinct n-grams in a dataset divided by the total number of n-grams. Score of 1 indicates that each n-gram occurs once in the dataset.

**Inverse Flesch Reading Ease** The Flesch Reading Ease (FRE) formula grades text from 100 to 0, 100 indicating most readable and 0 indicating difficult to read (Kincaid et al., 1975). We take the reciprocal of this measure.

**N-Gram and Character Diversity** Using the Shannon Index and Equitability described by Shannon (2001) we calculate the diversity and equitability of n-grams and characters. Probability distributions are count-based as described at the start of this section.

### 2.3 Models

To ensure that any discovered measures did not depend on which model was used (i.e. that they were model agnostic), we trained 12 models on every dataset. The models are summarised in Table 2. Hyperparameters were not optimised and were identical across all datasets. Specific implementation details of the models are described in Appendix C. Models were evaluated using the macro F1-Score. These models used three different representations of text to learn from to ensure that the discovered difficulty measure did not depend on the representation. These are:

**Word Embeddings** Our neural network models excluding the Convolutional Neural Network (CNN) used 128-dimensional FastText (Bojanowski et al., 2016) embeddings trained on the One Billion Word corpus (Chelba et al., 2013) which provided an open vocabulary across the datasets.

**Term Frequency Inverse Document Frequency (tf-idf)** Our classical machine learning models represented each data item as a tf-idf vector (Ramos et al., 2003). This vector has one entry for each word in the vocab and if a word occurs in a data item, then that position in the vector is the word's tf-idf score.

**Characters** Our CNN, inspired by Zhang et al. (2015), sees only the characters of each data item. Each character is assigned an ID and the list of IDs is fed into the network.

### 2.4 Genetic Algorithm

The genetic algorithm maintains a *population* of candidate difficulty measures, each being a binary vector of length 48 (see start of Method section). At each time step, it will evaluate each member of the population using a *fitness function*. It will then select pairs of parents based on their fitness, and perform *crossover* and *mutation* on each pair to produce a new child difficulty measure, which is added to the next population. This process is iterated until the fitness in the population no longer improves.

**Population** The genetic algorithm is non-randomly initialised with the 48 statistics described in Section 2.2 - each one is a difficulty measure composed of a single statistic. 400 pairs of parents are sampled with replacement from each population, so populations after this first time step will consist of 200 candidate measures. The probability of a measure being selected as a parent is proportional to its fitness.

**Fitness Function** The fitness function of each difficulty measure is based on the Pearson correlation (Benesty et al., 2009). Firstly, the Pearson correlation between the difficulty measure and the model test set score is calculated for each individual model. The Harmonic mean of the correlations of each model is then taken, yielding the fitness of that difficulty measure. Harmonic mean is used because it is dominated by its lowest constituents,

| Word Embedding Based | tf-idf Based | Character Based |
|---|---|---|
| LSTM-RNN | Adaboost | 3 layer CNN |
| GRU-RNN | Gaussian Naive Bayes (GNB) | - |
| Bidirectional LSTM-RNN | 5-Nearest Neighbors | - |
| Bidirectional GRU-RNN | (Multinomial) Logistic Regression | - |
| Multilayer Perceptron (MLP) | Random Forest | - |
| - | Support Vector Machine | - |

Table 2: Models summary organised by which input type they use.

so if it is high then correlation must be high for every model.

**Crossover and Mutation** To produce a new difficulty measure from two parents, the constituent statistics of each parent are randomly intermingled, allowing each parent to pass on information about the search space. This is done in the following way: for each of the 48 statistics, one of the two parents is randomly selected and if the parent uses that statistic, the child also does. This produces a child which has features of both parents. To introduce more stochasticity to the process and ensure that the algorithm does not get trapped in a local minima of fitness, the child is mutated. Mutation is performed by randomly adding or taking away each of the 48 statistics with probability 0.01. After this process, the child difficulty measure is added to the new population.

**Training** The process of calculating fitness, selecting parents and creating child difficulty measures is iterated until there has been no improvement in fitness for 15 generations. Due to the stochasticity in the process, we run the whole evolution 50 times. We run 11 different variants of this evolution, leaving out different statistics of the dataset each time to test which are most important in finding a good difficulty measure, in total running 550 evolutions. Training time is fast, averaging 79 seconds per evolution with a standard deviation of 25 seconds, determined over 50 runs of the algorithm on a single CPU.

## 3 Results and Discussion

The four hypothesized areas of difficulty - Class Diversity, Balance and Interference and Data Complexity - combined give a model agnostic measure of difficulty. All runs of the genetic algorithm produced different combinations of statistics which had strong negative correlation with model scores on the 78 datasets. The mean correlation was $-0.8795$ and the standard deviation was $0.0046$. Of the measures found through evolution we present two of particular interest:

1. **D1:** *Distinct Unigrams : Total Unigrams + Class Imbalance + Class Diversity + Top 5-Gram Interference + Maximum Unigram Hellinger Similarity + Unigram Mutual Info*. This measure achieves the highest correlation of all measures at $-0.8845$.

2. **D2:** *Distinct Unigrams : Total Unigrams + Class Imbalance + Class Diversity + Maximum Unigram Hellinger Similarity + Unigram Mutual Info*. This measure is the shortest measure which achieves a higher correlation than the mean, at $-0.8814$. This measure is plotted against model F1 scores in Figure 1.

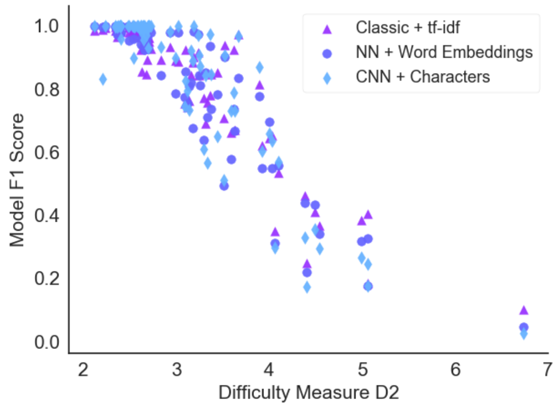

Figure 1: Model F1 scores against difficulty measure D2 for each of the three input types.

We perform detailed analysis on difficulty measure D2 because it relies only on the words of the dataset and requires just five statistics. This simplicity makes it interpretable and fast to calculate. All difficulty measures which achieved a correlation better than $-0.88$ are listed in Appendix D, where Figure 3 also visualises how often each metric was selected.

| Model | AG | Sogou | Yelp P. | Yelp F. | DBP | Yah A. | Amz. P. | Amz. F. | Corr. |
|---|---|---|---|---|---|---|---|---|---|
| **D2** | **3.29** | **3.77** | **3.59** | **4.42** | **3.50** | **4.51** | **3.29** | **4.32** | - |
| char-CNN (Zhang et al., 2015) | 87.2 | 95.1 | 94.7 | 62 | 98.3 | 71.21 | 95.1 | 59.6 | -0.86 |
| Bag of Words (Zhang et al., 2015) | 88.8 | 92.9 | 92.2 | 57.9 | 96.6 | 68.9 | 90.4 | 54.6 | -0.87 |
| Discrim. LSTM (Yogatama et al., 2017) | 92.1 | 94.9 | 92.6 | 59.6 | 98.7 | 73.7 | - | - | -0.87 |
| Genertv. LSTM (Yogatama et al., 2017) | 90.6 | 90.3 | 88.2 | 52.7 | 95.4 | 69.3 | - | - | -0.88 |
| Kneser-Ney Bayes (Yogatama et al., 2017) | 89.3 | 94.6 | 81.8 | 41.7 | 95.4 | 69.3 | - | - | -0.79 |
| FastText Lin. Class. (Joulin et al., 2016) | 91.5 | 93.9 | 93.8 | 60.4 | 98.1 | 72 | 91.2 | 55.8 | -0.86 |
| Char CRNN (Xiao and Cho, 2016) | 91.4 | 95.2 | 94.5 | 61.8 | 98.6 | 71.7 | 94.1 | 59.2 | -0.88 |
| VDCNN (Conneau et al., 2017) | 91.3 | 96.8 | 95.7 | 64.7 | 98.7 | 73.4 | 95.7 | 63 | -0.88 |
| **Harmonic Mean** | | | | | | | | | **-0.86** |

Table 3: Difficulty measure D2 compared to recent results from papers on large-scale text classification. The correlation column reports the correlation between difficulty measure D2 and the model scores for that row.

## 3.1 Does it Generalise?

A difficulty measure is useful as an analysis and performance estimation tool if it is model agnostic and provides an accurate difficulty estimate on unseen datasets.

When running the evolution, the F1 scores of our character-level CNN were not observed by the genetic algorithm. If the discovered difficulty measure still correlated with the CNN's scores despite never having seen them during evolution, it is more likely to be model agnostic. The CNN has a different model architecture to the other models and has a different input type which encodes no prior knowledge (as word embeddings do) or contextual information about the dataset (as tf-idf does). D1 has a correlation of $-0.9010$ with the CNN and D2 has a correlation of $-0.8974$ which suggests that both of our presented measures do not depend on what model was used.

One of the limitations of our method was that our models never saw text that was longer than 100 words and were never trained on any very large datasets (i.e. $>$1 million data points). We also performed no hyperparameter optimisation and did not use state-of-the-art models. To test whether our measure generalises to large datasets with text longer than 100 words, we compared it to some recent state-of-the-art results in text classification using the eight datasets described by Zhang et al. (2015). These results are presented in Table 3 and highlight several important findings.

**The Difficulty Measure Generalises to Very Large Datasets and Long Data Items.** The smallest of the eight datasets described by Zhang et al. (2015) has 120 000 data points and the largest has 3.6 million. As D2 still has a strong negative correlation with model score on these datasets, it seems to generalise to large datasets. Furthermore, these large datasets do not have an upper limit of data item length (the mean data item length in Yahoo Answers is 520 words), yet D2 still has strong negative correlation with model score, showing that it does not depend on data item length.

**The Difficulty Measure is Model and Input Type Agnostic.** The state-of-the-art models presented in Table 3 have undergone hyperparameter optimisation and use different input types including per-word learned embeddings (Yogatama et al., 2017), n-grams, characters and n-gram embeddings (Joulin et al., 2016). As D2 still has a strong negative correlation with these models' scores, we can conclude that it has accurately measured the difficulty of a dataset in a way that is useful regardless of which model is used.

**The Difficulty Measure Lacks Precision.** The average score achieved on the Yahoo Answers dataset is 69.9% and its difficulty is 4.51. The average score achieved on Yelp Full is 56.8%, 13.1% less than Yahoo Answers and its difficulty is 4.42. In ML terms, a difference of 13% is significant yet our difficulty measure assigns a higher difficulty to the easier dataset. However, Yahoo Answers, Yelp Full and Amazon Full, the only three of Zhang et al. (2015)'s datasets for which the state-of-the-art is less than 90%, all have difficulty scores $> 4$, whereas the five datasets with scores $> 90\%$ all have difficulty scores between 3 and 4. This indicates that the difficulty measure in its current incarnation may be more effective at assigning a class of difficulty to datasets, rather than a regression-like value.

## 3.2 Difficulty Measure as an Analysis Tool

| Statistic | Mean | Sigma |
|---|---|---|
| Distinct Words : Total Words | 0.0666 | 0.0528 |
| Class Imbalance | 0.503 | 0.365 |
| Class Diversity | 0.905 | 0.759 |
| Max. Unigram Hellinger Similarity | 0.554 | 0.165 |
| Top Unigram Mutual Info | 1.23 | 0.430 |

Table 4: Means and standard deviations of the constituent statistics of difficulty measure D2 across the 78 datasets from this paper and the eight datasets from Zhang et al. (2015).

As our difficulty measure has no dependence on learned weightings or complex combinations of statistics - only addition - it can be used to analyse the sources of difficulty in a dataset directly. To demonstrate, consider the following dataset:

**Stanford Sentiment Treebank Binary Classification (SST_2) (Socher et al., 2013b)** SST is a dataset of movie reviews for which the task is to classify the sentiment of each review. The current state-of-the-art accuracy is 91.8% (Radford et al., 2017).

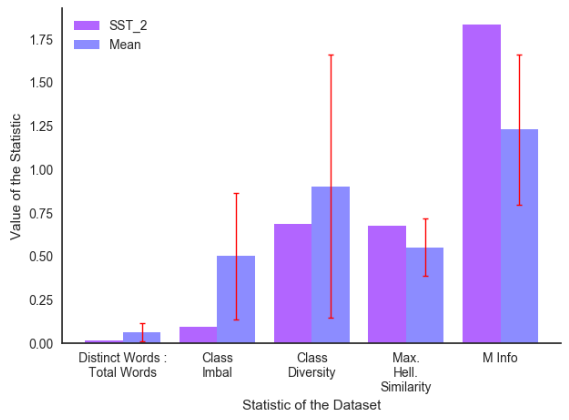

Figure 2: Constituents of difficulty measure D2 for SST, compared to the mean across all datasets.

Figure 2 shows the values of the constituent statistics of difficulty measure D2 for SST and the mean values across all datasets. The mean (right bar) also includes an error bar showing the standard deviation of statistic values. The exact values of the means and standard deviations for each statistic in measure D2 are shown in Table 4.

Figure 2 shows that for SST_2 the Mutual Information is more than one standard deviation higher than the mean. A high mutual information score indicates that reviews have both positive and negative features. For example, consider this review: *”de niro and mcdormand give solid performances but their screen time is sabotaged by the story ~~s~~ inability to create interest”* which is labelled ”positive”. There is a positive feature referring to the actors’ performances and a negative one referring to the plot. A solution to this would be to treat the classification as a multi-label problem where each item can have more than one class, although this would require that the data be relabelled by hand. An alternate solution would be to split reviews like this into two separate ones: one with the positive component and one with the negative.

Furthermore, Figure 2 shows that the Max. Hellinger Similarity is higher than average for SST_2, indicating that the two classes use similar words. Sarcastic reviews use positive words to convey a negative sentiment (Maynard and Greenwood, 2014) and could contribute to this higher value, as could mislabelled items of data. Both of these things portray one class with features of the other - sarcasm by using positive words with a negative tone and noise because positive examples are labelled as negative and vice versa. This kind of difficulty can be most effectively reduced by filtering noise (Smith et al., 2014).

To show that our analysis with this difficulty measure was accurately observing the difficulty in SST, we randomly sampled and analysed 100 misclassified data points from SST’s test set out of 150 total misclassified. Of these 100, 48 were reviews with both strong positive and negative features and would be difficult for a human to classify, 22 were sarcastic and 8 were mislabelled. The remaining 22 could be easily classified by a human and are misclassified due to errors in the model rather than the data items themselves being difficult to interpret. These findings show that our difficulty measure correctly determined the source of difficulty in SST because 78% of the errors are implied by our difficulty measure and the remaining 22% are due to errors in the model itself, not difficulty in the dataset.

### 3.3 The Important Areas of Difficulty

We hypothesized that the difficulty of a dataset would be determined by four areas not including noise: Class Diversity, Class Balance, Class Interference and Text Complexity. We performed multiple runs of the genetic algorithm, leaving statistics out each time to test which were most important in finding a good difficulty measure which resulted in the following findings:

**No Single Characteristic Describes Difficulty** When the Class Diversity statistic was left out of evolution, the highest achieved correlation was $-0.806$, $9\%$ lower than D1 and D2. However, on its own Class Diversity had a correlation of $-0.644$ with model performance. Clearly, Class Diversity is necessary but not sufficient to estimate dataset difficulty. Furthermore, when all measures of Class Diversity and Balance were excluded, the highest achieved correlation was $-0.733$ and when all measures of Class Interference were excluded the best correlation was $-0.727$. These three expected areas of difficulty - Class Diversity, Balance and Interference - must all be measured to get an accurate estimate of difficulty because excluding any of them significantly damages the correlation that can be found. Correlations for each individual statistic are in Table 6, in Appendix D.

**Data Complexity Has Little Affect on Difficulty** Excluding all measures of Data Complexity from evolution yielded an average correlation of $-0.869$, only $1\%$ lower than the average when all statistics were included. Furthermore, the only measure of Data Complexity present in D1 and D2 is Distinct Words : Total Words which has a mean value of $0.067$ and therefore contributes very little to the difficulty measure. This shows that while Data Complexity is necessary to achieve top correlation, its significance is minimal in comparison to the other areas of difficulty.

### 3.4 Error Analysis

#### 3.4.1 Overpowering Class Diversity

When a dataset has a large number of balanced classes, then Class Diversity dominates the measure. This means that the difficulty measure is not a useful performance estimator for such datasets.

To illustrate this, we created several fake datasets with 1000, 100, 50 and 25 classes. Each dataset had 1000 copies of the same randomly generated string in each class. It was easy for models to overfit and score a $100\%$ F1 score on these fake datasets.

For the 1000-class fake data, Class Diversity is $6.91$, which by our difficulty measure would indicate that the dataset is extremely difficult. However, all models easily achieve a $100\%$ F1 score. By testing on these fake datasets, we found that the limit for the number of classes before Class Diversity dominates the difficulty measure and renders it inaccurate is approximately 25. Any datasets with more than 25 classes with an approximately equal number of items per class will be predicted as difficult regardless of whether they actually are because of this diversity measure.

Datasets with more than 25 *unbalanced* classes are still measured accurately. For example, the ATIS dataset (Price, 1990) has 26 classes but because some of them have only 1 or 2 data items, it is not dominated by Class Diversity. Even when the difficulty measure is dominated by Class Diversity, examining the components of the difficulty measure independently would still be useful as an analysis tool.

#### 3.4.2 Exclusion of Useful Statistics

One of our datasets of New Year's Resolution Tweets has 115 classes but only 3507 data points (FigureEight, 2018). An ML practitioner knows from the number of classes and data points alone that this is likely to be a difficult dataset for an ML model.

Our genetic algorithm, based on an unweighted, linear sum, cannot take statistics like data size into account currently because they do not have a convenient range of values; the number of data points in a dataset can vary from several hundred to several million. However, the information is still useful to practitioners in diagnosing the difficulty of a dataset.

Given that the difficulty measure lacks precision and may be better suited to classification than regression as discussed in Section 3.1, cannot take account of statistics without a convenient range of values and that the difficulty measure must be interpretable, we suggest that future work could look at combining statistics with a white-box, non-linear algorithm like a decision tree. As opposed to summation, such a combination could take account of statistics with different value ranges and perform either classification or regression while remaining interpretable.

### 3.5 How to Reduce the Difficulty Measure

Here we present some general guidelines on how the four areas of difficulty can be reduced.

Class Diversity can only be sensibly reduced by lowering the number of classes, for example by grouping classes under superclasses. In academic settings where this is not possible, hierarchical learning allows grouping of classes but will produce granular labels at the lowest level (Kowsari et al., 2017). Ensuring a large quantity of data in each class will also help models to better learn the features of each class.

Class Interference is influenced by the amount of noise in the data and linguistic phenomena like sarcasm. It can also be affected by the way the data is labelled, for example as shown in Section 3.2 where SST has data points with both positive and negative features but only a single label. Filtering noise, restructuring or relabelling ambiguous data points and detecting phenomena like sarcasm will help to reduce class interference. Easily confused classes can also be grouped under one superclass if practitioners are willing to sacrifice granularity to gain performance.

Class Imbalance can be addressed with data augmentation such as thesaurus based methods (Zhang et al., 2015) or word embedding perturbation (Zhang and Yang, 2018). Under- and oversampling can also be utilised (Chawla et al., 2002) or more data gathered. Another option is transfer learning where knowledge from high data domains can be transferred to those with little data (Jaech et al., 2016).

Data Complexity can be managed with large amounts of data. This need not necessarily be labelled - unsupervised pre-training can help models understand the form of complex data before attempting to use it (Halevy et al., 2009). Curriculum learning may also have a similar effect to pre-training (Bengio et al., 2009).

### 3.6 Other Applications of the Measure

**Model Selection** Once the difficulty of a dataset has been calculated, a practitioner can use this to decide whether they need a complex or simple model to learn the data.

**Performance Checking and Prediction** Practitioners will be able to compare the results their models get to the scores of other models on datasets of an equivalent difficulty. If their models achieve lower results than what is expected according to the difficulty measure, then this could indicate a problem with the model.

## 4 Conclusion

When their models do not achieve good results, ML practitioners could potentially calculate thousands of statistics to see what aspects of their datasets are stopping their models from learning. Given this, how do practitioners tell which statistics are the most useful to calculate? Which ones will tell them the most? What changes could they make which will produce the biggest increase in model performance?

In this work, we have presented two measures of text classification dataset difficulty which can be used as analysis tools and performance estimators. We have shown that these measures generalise to unseen datasets. Our recommended measure can be calculated simply by counting the words and labels of a dataset and is formed by adding five different, unweighted statistics together. As the difficulty measure is an unweighted sum, its components can be examined individually to analyse the sources of difficulty in a dataset.

There are two main benefits to this difficulty measure. Firstly, it will reduce the time that practitioners need to spend analysing their data in order to improve model scores. As we have demonstrated which statistics are most indicative of dataset difficulty, practitioners need only calculate these to discover the sources of difficulty in their data. Secondly, the difficulty measure can be used as a performance estimator. When practitioners approach new tasks they need only calculate these simple statistics in order to estimate how well models are likely to perform.

Furthermore, this work has shown that for text classification the areas of Class Diversity, Balance and Interference are essential to measure in order to understand difficulty. Data Complexity is also important, but to a lesser extent.

Future work should firstly experiment with non-linear but interpretable methods of combining statistics into a difficulty measure such as decision trees. Furthermore, it should apply this difficulty measure to other NLP tasks that may require deeper linguistic knowledge than text classification, such as named entity recognition and parsing. Such tasks may require more advanced features than simple word counts as were used in this work.

## References

Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. 2015. Tubespam: Comment spam filtering on youtube. In *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*, pages 138–143. IEEE. [Online; accessed 22-Feb-2018].

Tiago A. Almeida and Jos Mara Gmez Hidalgo. 2011. Sms spam collection v. 1. [Online; accessed 25-Feb-2018].

Ioannis Antonellis, Christos Bouras, and Vassilis Poulopoulos. 2006. Personalized news categorization through scalable text classification. In *Asia-Pacific Web Conference*, pages 391–401. Springer.

archive.ics.uci.edu. 2018. Sentence classification data set. `https://archive.ics.uci.edu/ml/datasets/Sentence+Classification`. [Online; accessed 20-Feb-2018].

Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM.

Ester Bernadó-Mansilla and Tin Kam Ho. 2005. Domain of competence of xcs classifier system in complexity measurement space. *IEEE Transactions on Evolutionary Computation*, 9(1):82–104.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. *arXiv preprint arXiv:1702.08303*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Carla E Brodley and Mark A Friedl. 1999. Identifying mislabeled training data. *Journal of artificial intelligence research*, 11:131–167.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Ed Collins, Isabelle Augenstein, and Sebastian Riedel. 2017. A supervised approach to extractive summarisation of scientific papers. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 195–205.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1107–1116.

Alice Coucke. 2017. Benchmarking natural language understanding systems: Google, facebook, microsoft, amazon, and snips. [Online; accessed 7-Feb-2018].

Thomas M Cover and Joy A Thomas. 2012. *Elements of information theory*. John Wiley & Sons.

Datafiniti. 2018. Grammar and online product reviews. [Online; accessed 26-Feb-2018].

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

D. A. Elizondo, R. Birkenhead, M. Gamez, N. Garcia, and E. Alfaro. 2009. Estimation of classification complexity. In *2009 International Joint Conference on Neural Networks*, pages 764–770.

FigureEight. 2018. Data for everyone. `https://www.figure-eight.com/data-for-everyone/`. [Online; accessed 25-Feb-2018].

Maya R Gupta, Samy Bengio, and Jason Weston. 2014. Training highly multiclass classifiers. *The Journal of Machine Learning Research*, 15(1):1461–1492.

Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12.

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. 2009. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360.

Tin Kam Ho and M. Basu. 2000. Measuring the complexity of classification problems. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 2, pages 43–47 vol.2.

Tin Kam Ho and M. Basu. 2002. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300.

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

Jin-Hyuk Hong and Sung-Bae Cho. 2008. A probabilistic multi-class strategy of one-vs.-rest support vector machines for cancer classification. *Neurocomputing*, 71(16-18):3275–3281.

Paul Jaccard. 1912. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50.

Aaron Jaech, Larry Heck, and Mari Ostendorf. 2016. Domain adaptation of recurrent neural networks for natural language understanding. *arXiv preprint arXiv:1604.00117*.

Nathalie Japkowicz. 2000. The class imbalance problem: Significance and strategies. In *Proc. of the Intl Conf. on Artificial Intelligence*.

Licheng Jiao and Lei Wang. 2000. A novel genetic algorithm based on immunity. *IEEE Transactions on Systems, Man, and Cybernetics-part A: systems and humans*, 30(5):552–561.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. ACM.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*, pages 364–371. IEEE.

Lucien Le Cam and Grace Lo Yang. 2012. *Asymptotics in statistics: some basic concepts*. Springer Science & Business Media.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

N. Maci, A. Orriols-Puig, and E. Bernad-Mansilla. 2008. Genetic-based synthetic data sets for the analysis of classifiers behavior. In *2008 Eighth International Conference on Hybrid Intelligent Systems*, pages 507–512.

E. B. Mansilla and Tin Kam Ho. 2004. On classifier domains of competence. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 136–139 Vol.1.

Diana Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *Lrec*, pages 4238–4243.

G Harry Mc Laughlin. 1969. Smog grading-a new readability formula. *Journal of reading*, 12(8):639–646.

David Martin Powers. 2011. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

Patti J Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990.*

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142.

Jason D Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 616–623.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

RJ Senter and Edgar A Smith. 1967. Automated readability index. Technical report, CINCINNATI UNIV OH.

Claude Elwood Shannon. 2001. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. 2014. The potential benefits of filtering versus hyper-parameter optimization. *arXiv preprint arXiv:1403.3342*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013a. 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.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013b. 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.

Darrell Whitley. 1994. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85.

wiki.dbpedia.org. 2015. Data set 2.0. `http://wiki.dbpedia.org/data-set-20`. [Online; accessed 21-Feb-2018].

Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv preprint arXiv:1602.00367*.

Yelp.com. 2018. Yelp dataset challenge. `http://www.yelp.com/dataset_challenge`. [Online; accessed 23-Feb-2018].

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*.

Dongxu Zhang and Zhichao Yang. 2018. Word embedding perturbation for sentence classification. *arXiv preprint arXiv:1804.08166*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.