Multilateral-Powered Social Media Classification

Brian Morales & Dr. Manuel E. Lladser
Department of Applied Mathematics
University of Colorado Boulder

1 Introduction

In a world full of data one area of complication is Natural Language Processing (NLP). NLP is the study of machines understanding human language and it is important for sentiment analysis, translation, and bot detection. In social media, people are constantly posting how they feel, what's happening in their life, and opinionated ideas. We would like to apply NLP to distinguish different social media posts.

There are many popular sentiment analysis models that achieve high accuracy of classification. To name a few advanced sentiment models there exist Word2Vec, GloVe, BERT, or FastText, all constructing an embedding to represent input text. These sentiment algorithms are trained on deep learning models such as Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), and Recursive Neural Networks (RNN). However, the issue with these deep learning models is they achieve high accuracy at the cost of unexplained performance. Similar to a black box that accepts inputs, and outputs a correct answer without knowing how it achieved that answer.

In this paper, we introduce a new embedding for numerically encoding and classifying sentences, or, as we like to call it, bags of words. I will explain how we treat our bags of words as a set of landmarks, find unique distances between these landmarks, and associate positive, negative, and neutral words around similar distant landmarks.

We will first discuss multilateration/resolving sets, a metric dimension that finds unique sets. Then Jaccard distance, a distance metric that is nonnegative, symmetric, and satisfies the triangular inequality. After, the Information Context Heuristic (ICH) algorithm, finds nearly optimal resolving sets in a finite metric space utilizing a distance metric. Our probabilistic approach, which circumvents the infeasibility of the ICH. Lastly, embedding our bag of words to find optimal sets that can cluster positive, negative, and neutral words. We will then test our new embedding with Support Vector Classifiers (SVC) to observe how well our embedding performed.

1.1 Problem Space

Our problem space is Twitter posts and we utilized the Valence Aware Dictionary and Sentiment Reasoner (VADER) dataset. VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media (Hutto and Gilbert, 2014). We use the VADER Sentiment Analysis dataset to synthetically test our embedding. The VADER dataset is fully open-sourced under the MIT License.

We are using the 'Full_Vader' dataset which contains sentiment-expressed words that have been classified as either positive, negative, or neutral. The data was obtained from a GitHub repository that was created by professional linguists. We use the dataset to synthetically create sentences according to tones and test the application of multilateration and Jaccard distances.

2 Approach

2.1 Method Outline

This paper has two objectives: to apply our embedding to the VADER dataset and test its efficiency with SVCs. Much of the SVC analysis compares cross-contaminated bags of words, for example, bags of words that are %90 positive and %10 negative. Noise-contaminated bags of words, a set of words that contain neutral words. And both, noise and cross-contaminated bags of words.

Let X be our dataset of words, where $X = 1050$. We represent English sentences as bags of words that are elements of the power set

of X, denoted as $2^{|X|}$. For instance, let $A = \{queen, royal\}$, then the power set of A is:

$$P(A) = \{\{\emptyset\}, \{queen\}, \{royal\}, \{queen, royal\}\}$$

which is of size $2^{|A|} = 2^2 = 4$. Here we denote $|A|$ as the cardinality of A. Now that we defined our bags of words to implement our embedding we need two important tools: Jaccard distance and multilateration.

2.2 Jaccard Distance

We would like to find a distance metric, $d(A, B)$ such that it satisfies nonnegativity, symmetry, and triangular inequality (Lladser and Paradise,). We establish X with the so-called Jaccard Distance, which is defined for all $A, B \in 2^X$ as follows:

$$d(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

where $A \cap B$ is defined as the intersection and $A \cup B$ is the union of sets A and B. This follows the presumptions that its nonnegative, symmetric, and satisfies the triangular inequality. For example, lets $A = \{queen, royal, god\}$ and $B = \{queen, king, god\}$. The *Jaccard Distance* between sets A and B is:

$$1 - \frac{|\{queen, god\}|}{|\{queen, royal, king, god\}|} = 1 - \frac{2}{4} = \frac{1}{2}$$

In this paper, we want to characterize the metric dimension of $(2^X, d)$ where 2^X is our English bags of words and d is the Jaccard distance. We want to find the smallest possible size of $R \subset 2^X$ with the following property: if $a \in 2^X$ and $\forall b \in 2^X$ excluding a, are such that $d(a, r) \neq d(b, r)$ for all $r \in R$, then $d(a, r)$ has a unique distance. In other words, if we think of the elements in R as "landmarks" in 2^X, then any subset of X has a unique Jaccard Distance to those landmarks. We denote the metric dimension of 2^X with respect to the Jaccard distance as $\beta(2^X)$ (Lladser and Paradise,).

The idea of metric dimension is thought of as an abstraction of the idea of trilateration of the Euclidean plane to metric spaces. In our case, however, we will have multiple "landmarks" or an arbitrary metric space. We will need to multilaterate our English bags of words. Hence, we apply the Jaccard distance to find a set, R, that can uniquely represent every bag of words by k-dimensional vector of distance.

2.3 Multilateration

With the Jaccard Distance we would like to multilaterate or, resolve, our space 2^X to find the minimum size R such that the bags of words in R all have unique distances for all elements in 2^X. We define a resolving set as follows: $R = \{r_1, ..., r_k\} \subset 2^X$ (i.e. a collection of subsets of X) resolves the metric space $(2^X, d)$ if every $A \in 2^X$ can be uniquely represented by the k-dimensional vector of distance (Tillquist et al.,):

$$(d(A, r_1), ..., d(A, r_k))$$

For instance, if $Y = \{1, 2, 3\}$, then $R = \{\{1, 3\}, \{2, 3\}\}$ resolves 2^Y because no two rows are identical over the last two columns, as shown in the table below.

Jaccard distance	$\{1,3\}$	$\{2,3\}$
$\emptyset$	1	1
$\{1\}$	1/2	1
$\{2\}$	1	1/2
$\{3\}$	1/2	1/2
$\{1,2\}$	2/3	2/3
$\{1,3\}$	0	2/3
$\{2,3\}$	2/3	0
$\{1,2,3\}$	1/3	1/3

In this example, we treat R like (x, y) coordinates in Euclidean space. Notice that each row does have the exact same coordinates as any other row. In our examples, R is the set of "landmarks" and each element in the space 2^Y has a unique distance to our set of "landmarks".

Applying this result to our space 2^X of English sentencess, we would like to find a minimum set R, or "landmarks", that will help us separate each element in 2^X by their unique Jaccard Distance. Particularly, if a set of bags of words contain similar tones or root words then their Jaccard Distance to the set of "landmarks" will be similar, but not exactly the same. Therefore, positive words will all have similar distances to the set of "landmarks", and similarly with negative words.

To find resolving sets we initially adopted the Information Context Heuristic (ICH) algorithm, however, as we discuss further the ICH proved to be infeasible with the size of our dataset. In the next section we will briefly discuss the ICH algorithm and our solution to circumvent the ICH.

2.4 Information Context Heuristic (ICH) and Probabilistic Approach

In our example in multilateration, we found our resolving set, R, utilizing the ICH. In principle, the ICH algorithm can find nearly optimal resolving sets in finite metric spaces. Unfortunately, the ICH algorithm requires the distance matrix, in our case the Jaccard distance matrix $d(A, B)$ with $A, B \in 2^X$, to be stored in computer memory. Our dataset of words is of size $|X| = 1050$, now 2^X is a very large value and the distance matrix of $2^X \times 2^X$ is too large for our computer to store. Therefore the ICH becomes rapidly infeasible as $|X|$ increases. Our solution was a probabilistic approach.

We construct a random set $R = \{r_1, ..., r_k\}$ where each word in our dataset X is added to the set r_i with a probability of $\frac{1}{2}$. Then for all X large enough and constant $C > 0, R = \{r_i, ..., r_k\}$ with

$$k \sim \frac{C \cdot |X|}{ln|X|}$$

resolves all bags of words of equal size with overwhelmingly high probability. The probability of two sets of different sizes colliding is asymptotically negligible. In short, the probabilistic approach means if the size of R is greater than the ratio shown above, this means R resolves, and we are able to multilaterate our bags of words.

Bringing this all together, our method first finds a resolving set of size $|R|$ greater than our probabilistic approach ratio. Then we multilaterate our bag of words, 2^X, such that each bag of words is a unique vector of distances in a high dimensional point. In our dataset, each bag of words is represented as a 325 dimensional vector of Jaccard distances to sets in R, or set of "landmarks." To test the performance of our embedding we utilize Support Vector Classifiers.

2.5 Support Vector Classifiers (SVC)

The support vector classifier, sometimes called a soft margin classifier, is an extension of a simple and intuitive classifier called the maximal marginal classifier. A support vector classifier is a model that uses hyperplanes to separate data points into different classes. The support vector classifier is a natural approach for classification in the two-class, three-class setting if the boundary between the two classes is linear. Rather than seeking the largest possible hyperplane so that every observation is not only on the correct side of the hyperplane, we instead allow some observations to be on the incorrect side of the hyperplane. We applied a support vector classifier model to test the performance of our embedding.

3 Results

Since our dataset set consists of a three-tone class, support vector classifiers were optimal for our choice of performance. Each bag of words is represented as a 325 dimensional vector of Jaccard distances with the size of each sentence drawn from a Poisson distribution with rate $\lambda = 50$. We obtain 325 from our probabilistic approach, where $|R| = 325$ was the smallest size to obtain resolving sets. We pass 325 features to our support vector classifier and split our data into 70% training and 30% testing.

3.1 PCA Visualization

We utilized PCA to visualize the performance of our support vector classifiers. As seen in Fig. 1, we have a consistent separation between positive, negative, and neutral tones. In this experiment, pure data means bags of words that were assembled using one tone.

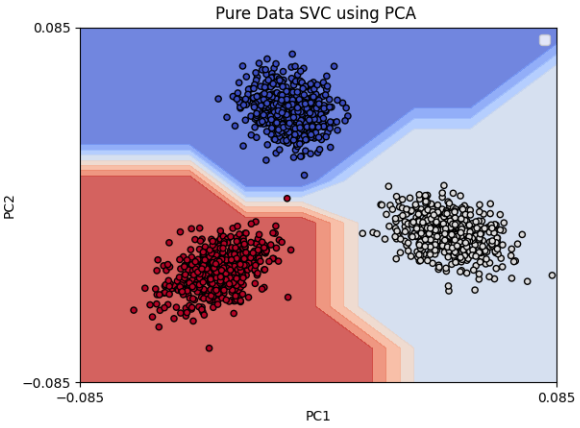

Figure 1: Pure data

For our pure data results, we obtained an accuracy of 99%. The synthetic sentences was projected onto PC1 and PC2. This accuracy was accordant with the full model (all 325 features) and the reduced model (PCA). Next, we wanted to see the performance with cross-contaminated sentences.

Cross-contaminated means that we mixed our bags of words with both positive and negative words. In Fig. 2 we plot a dual tone with 10% cross-contamination. The projection was onto PC1 and PC2 with an accuracy of 99%.

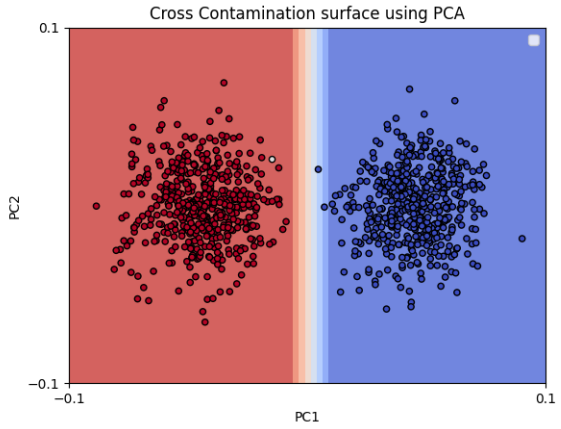

Figure 2: Pure data

3.2 Incrementing Features

Next, we wanted to observe how many features are actually needed to obtain satisfactory accuracy. Training on the pure data, from Fig. 3, we concluded that at least 70% of the features are needed to achieve an accuracy of 100%.

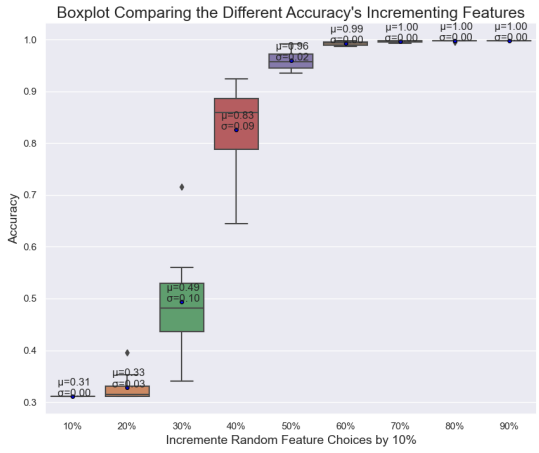

Figure 3: Average Features

The 70% was found by training and testing 10 times over the pure dataset every time we incremented the number of features by 10%. The increment of features was randomly selected every 10th iteration.

3.3 Addition of Noise

Following that, we wanted to examine how effective our embedding was when we added noise. We define noise as inserting neutral words in positive/negative bags of words. Similar to Fig. 3, we incremented the amount of noise by 5% every 10th iteration of the model.

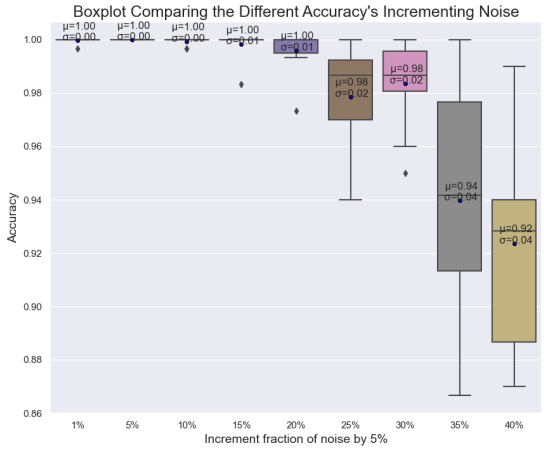

Figure 4: Average Noise

In Fig. 4, we see a downward trend as the amount of noise increases. However, the average accuracy when there is 40% of noise is $\mu = 92\%$. This signifies that our embedding performs well when we have large amounts of noise.

3.4 Cross-Contamination

Similarly, we analyzed how well our embedding performs when cross-contamination increases. In Fig. 5, there is a downward trend as cross-contamination increases, however, the average accuracy compounds as it decreases.

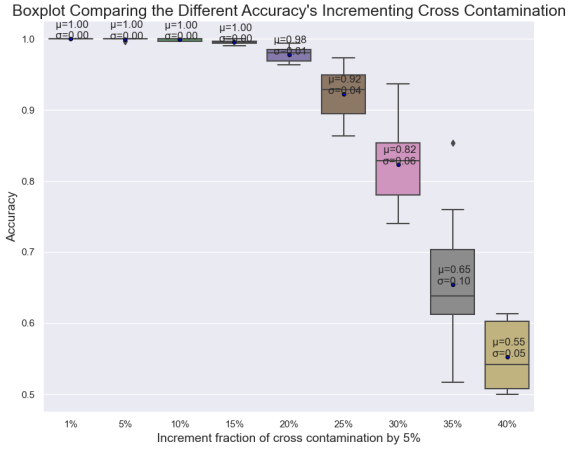

Figure 5: Average Cross-contamination

Unlike Fig. 4, when cross-contamination is at 40% our average accuracy is $\mu = 55\%$. This suggests that our embedding is underfitting when we increase our cross-contamination. Our accuracy becomes as good as flipping a coin. To

make our test more realistic we decided to add both noise and cross-contamination.

3.5 Cross-contamination & Noise

In Fig. 6 we test our embedding decreasing noise and increasing cross-contamination. We start with 20% noise and 15% cross-contamination until we reach 10% noise and 30% cross-contamination.

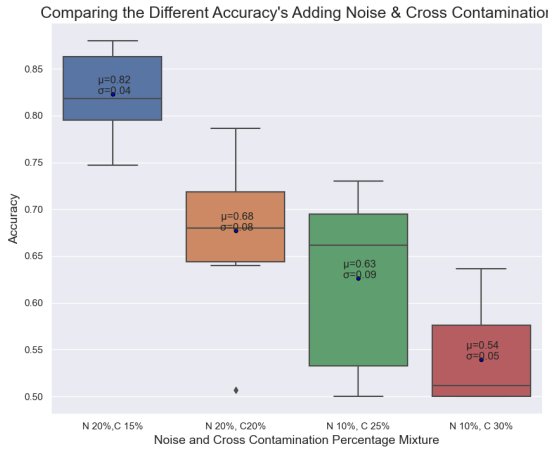

Figure 6: Average Cross-contamination & Noise

Like Fig. 5, the average accuracy when noise and cross-contamination are 10% and 30%, respectively, is as good as flipping a coin. We believe that this outcome is most likely affected by cross-contamination, hence, we would like to make our embedding more robust to cross-contamination.

4 Conclusion

We would like to apply our embedding to distinguish different social media posts, specifically tweets. In our work, we multilaterate our dataset, 2^X where X is our data set of words and 2^X represents English sentences as bags of words. We multilaterate the bags of words to find resolving sets, or landmarks, such that each sentence is a unique vector of Jaccard distances in a high dimensional point.

We test our embedding by leveraging synthetic bags of words and then labeling them with positive, negative, or neutral tones. We add noise and cross-contaminate to our synthetic sentences to note how robust are embedding is. Eventually, we would like to apply this to actual tweets.

What we would like to achieve is a differentiation of tweets in politics, cyberbullying, and threatening tweets. Yet, when it comes to human tweets there is grammatical error, different jargon, and dialect, and no use of a sentence at all. We have plans to preprocess our data and improve the embedding's robustness to cross-contamination and noise, as well as sentence length.

References

C.J. Hutto and E.E. Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. eighth international conference on weblogs and social media (icwsm-14), Jun.

Manuel E. Lladser and Alexander J. Paradise. *Metric Dimension of Jaccard Metric Spaces.*

Richard C. Tillquist, Rafael M. Frongillo, and Manuel E. Lladser. Getting the lay of the land in discrete space: A survey of metric dimension and its applications .