

Offensive Language Detection: A Comparative Analysis

Vyshnav M T*, Sachin Kumar S, Soman K P

*Center for Computational Engineering & Networking (CEN)
Amrita School of Engineering, Coimbatore.
Amrita Vishwa Vidyapeetham, India.*

Abstract

Offensive behaviour has become pervasive in the Internet community. Individuals take the advantage of anonymity in the cyber world and indulge in offensive communications which they may not consider in the real life. Governments, online communities, companies etc are investing into prevention of offensive behaviour content in social media. One of the most effective solution for tackling this enigmatic problem is the use of computational techniques to identify offensive content and take action. The current work focuses on detecting offensive language in English tweets. The dataset used for the experiment is obtained from SemEval-2019 Task 6 on Identifying and Categorizing Offensive Language in Social Media (OffensEval). The dataset contains 14,460 annotated English tweets. The present paper provides a comparative analysis and Random kitchen sink (RKS) based approach for offensive language detection. We explore the effectiveness of Google sentence encoder, Fasttext, Dynamic mode decomposition (DMD) based features and Random kitchen sink (RKS) method for offensive language detection. From the experiments and evaluation we observed that RKS with fasttext achieved competing results. The evaluation measures used are accuracy, precision, recall, f1-score.

Keywords: Offensive language, Fasttext, Google universal sentence encoder, Dynamic mode decomposition, Random kitchen sink, Support vector machines

1. Introduction

In this digital era, online discussions and interactions has become a vital part of daily life of which a huge part is covered by social media platforms like twitter, facebook, instagram etc. Similar to real life there exist anti-social elements in the cyberspace, who take advantage of the anonymous nature in cyber world and indulge in vulgar and offensive communications. This includes bullying, trolling, harassment [1, 2] and has become a growing concern for governments. Youth experiencing such victimization was recorded to have psychological symptoms of anxiety, depression, loneliness [2]. Thus it is important to identify and remove such behaviours at the earliest. One solution to this is the automatic detection using machine learning algorithms.

Detecting offensive language from social media is a challenging research problem due to the different level of ambiguities present in the natural language and noisy nature of the social media language. Moreover, social media subscribers are from linguistically diverse and varying communities. Overseeing the complication of this problem, [3] organized a task in SemEval2019, Task 6: Identifying and Categorizing Offensive Language in Social Media. The tweets were collected by the organizers using Twitter API and have annotated them in a hierarchical manner as offensive language present in the tweet, type of the offense and target of the offense. There were three sub-tasks according to the hierarchy of annotation: a) To detect if a post is offensive (OFF) or not (NOT), b) To Identify the type of offense in the post as targeted threat (TTH), targeted insult (TIN), untargeted (UNT), c) To identify if offense is targeted to organization or entity (ORG), group of people (GRP), individual (IND), or other (OTH).

*Corresponding author:

Email address: vyshnav94.mec@gmail.com,
kp_soman@amrita.edu (Soman K P)

The dataset had the following challenges:

- Dataset was comparatively smaller.

- Dataset was biased/imbalanced [4].

In this paper, we are proposing a comparative analysis for the sub-task A :Offensive language identification of the SemEval2019, Task 6. Sub-task A was the most popular sub-task among the three and had total 104 team participation. In Table 2, the list of first 5 participants along with system and f1-score has been shown.

Offensive language detection is one of the challenging and interesting topic for research. Recent past had multiple shared tasks and research on this topic. One of the initial work on offensive language using supervised classification was done by Yin et al. [9]. They have used Ngram, TFIDF and combination of TFIDF with Sentiment and Contextual as Features. Schmidt and Wiegand [10] gave a survey on automatic hate speech detection using NLP. The authors surveyed on features like Simple Surface Features, Word Generalization, Linguistic Features, Knowledge-Based Features, Multimodal Information etc. In 2013, a study on detecting cyberbullying in YouTube comments was done by Dadvar et al. [11]. They have used a combination of content-based, cyberbullying-specific and user-based features and showed that detection of cyberbullying can be improved by taking user context into account. Shared task GermEval 2018 organised by Wiegand et al.,[12] was focused on offensive language detection on German tweets. It had a dataset of over 8,500 annotated tweets and was trained for binary classification of offensive and non-offensive tweets. They obtained an overall macro-F1 score of 76.77%. Another shared task on Aggression Identification in Social Media was organised by Kumar et al., [13]. The task provided a dataset with 15,000 annotated Facebook posts and comments in Hindi and English. They obtained a weighted F-score of 64% for both English and Hindi. The rest of the paper is structured as follows. Section 2 explains about the methodology with formulation. Section 3 discusses on the Proposed approach. Section 4 talks on the Experiments and discussions performed. Finally, conclusion is given in Section 5.

2. Methodology

2.1. Data Pre-processing

Data pre-processing is a very crucial step which needs to be done before applying any machine learning tasks, because the real time data could be very noisy and unstructured. For the two models used in this work, pre-processing of tweets is done separately:

- **Pre-processing for Google model:**

It has become a common culture to use #tags across social media. So we have replaced multiple #tags with a single #tag. Mostly @ symbol is used to mention person or entities in a tweet. So we replace multiple @symbols with a single @-mention. Some tweets may contain the link to a website or some other urls. So we replace all of these with a single keyword URLS.

- **Pre-processing for fasttext model:**

For applying fasttext model to get word vectors, we followed a different set of pre-processing steps. First, all the numbers, punctuation marks, urls (http:// or www.) and symbols (emoji, #tags, -mention) were removed from the tweet as it do not contain information related to sentiment. After that, tokenization and lowercasing was applied to the tweets. Tokenization was done using tokenizer from NLTK package [18]. Finally, the stop word are removed. The list is obtained from NLTK package.

2.2. Embeddings

Word embeddings are ubiquitous for any NLP problem, as algorithms cannot process the plain text or strings in its raw form. Word embeddings are vectors that captures the semantic and contextual information of words. The word embedding used for this work are:

- **FastText:** The fastText algorithm created by Facebook [16] assumes every word to be n-grams of character. It helps to give the vector representations for out of vocabulary words. For the current work, fasttext based word embedding is used for generating token vectors of dimension 300 [14]. Each vector corresponding to the tweet is generated by taking the average of token vectors.
- **Universal Sentence Encoder:** Developed by Google, Universal sentence encoder [15, 17] provides embeddings at sentence level. The dimension of the embedding vector is 512, irrespective of the number of tokens in the input tweet. These vectors can capture good semantic information from the sentences. For each tweet, this model generates a 512 length embedding vector and is used as features for further classification.
- **DMD and HODMD:** DMD is a method initially used in fluid dynamics which captures spatio-temporal features [19]. It has been used in

background-foreground separation [20], load forecasting [21], saliency detection [22] etc. For natural language processing, DMD has been first applied for sentiment analysis [23, 24]. This motivated to explore DMD based feature for the present work.

Dynamic mode decomposition (DMD) is a much more powerful concept and it assumes the evolution of the function over the rectangular field is effected by the mapping of a constant matrix A . A captures the system's inherent dynamics and the aim of the DMD is to understand using its dominant eigenvalues and eigenvectors. Assumption is that this matrix A is of low rank and hence the sequence of vectors $x_1, x_2, x_3, \dots, x_k, \dots, x_{m+1}$ finally become a linearly dependent set. That is, vector x_{m+1} become linearly dependent on previous vectors. The data matrix X in terms of eigen vectors associated with matrix A .

$$A^k x_1 = \Phi \Lambda^k \Phi^\dagger x_1 = \Phi \Lambda^k b \quad (1)$$

$$A^k x_1 = \Phi \Lambda^k \Phi^\dagger x_1 = \Phi \Lambda^k b = x_{k+1} \quad (2)$$

where, Φ^\dagger is pseudo inverse of Φ . A is of rank m and Φ have m columns. Hence, pseudo-inverse will do the job than inverse operation. The columns of Φ are called DMD modes and this forms the features.

$$\begin{aligned} x_{k+1} &= \Phi \begin{pmatrix} \lambda_1^k b_1 \\ \lambda_2^k b_2 \\ \vdots \\ \lambda_m^k b_m \end{pmatrix} \\ &= \begin{pmatrix} \phi_1 & \phi_2 & \dots & \phi_m \end{pmatrix} \begin{pmatrix} \lambda_1^k b_1 \\ \lambda_2^k b_2 \\ \vdots \\ \lambda_m^k b_m \end{pmatrix} \end{aligned} \quad (3)$$

Time-lagged matrices are prepared as snapshot for this approach. In Eigensent [25], the authors proposed HODMD to find embeddings for sentences. The authors suggested, sentences can be represented as a signal using word embeddings by taking the average of word vectors. This is intuitive because word embeddings almost obeys the laws of linear algebra, by capturing word analogies and relationships. Therefore, by considering every sentence as a multi-dimensional signal, we can capture the important transitional dynamics of sentences. Also, for the signal representation of sentences,

each word vector will act as a single point in the signal. For the present work, to generate DMD and HODMD based features, Fastext based embedding is used.

3. Proposed Approach

RKS approach proposed in [26, 27], explicitly maps data vectors to a space where linear separation is possible. It has been explored for natural language processing tasks [29, 30]. The RKS method provides an approximate kernel function via explicit mapping.

$$K(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle \approx \langle Z(x_1), Z(x_2) \rangle \quad (4)$$

Here, $\phi(\cdot)$ denotes the implicit mapping function (used to compute kernel matrix), $Z(\cdot)$ denotes the explicit mapping function using RKS and Ω_k denotes random variable.

$$Z(x) = \sqrt{1/k} \begin{bmatrix} \cos(x^T \Omega_1) \\ \vdots \\ \cos(x^T \Omega_k) \\ \sin(x^T \Omega_1) \\ \vdots \\ \sin(x^T \Omega_k) \end{bmatrix} \quad (5)$$

Figure 1 show the block diagram of the proposed approach.

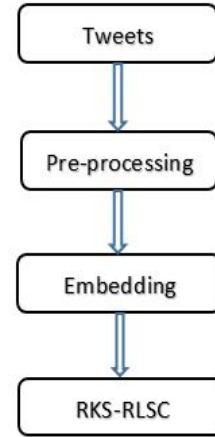


Figure 1: Illustrates the block diagram for the proposed approach

4. Experiments and Discussions

4.1. Data Description

OLID (Offensive Language Identification Dataset) is a collection of English tweets which are annotated using

a three-level hierarchical annotation model. It was collected using Twitter API and contains 14,460 annotated tweets. The task divided the data as train, trial and test, of which train and trial was initially released as starting kit, finally test was released as Test A release. All three of these partitions were highly biased, thus making the task more challenging and real time. The train set had 13,240 tweets, out of which 8840 tweets were not offensive (NOT) and 4400 tweets were offensive (OFF). Similarly, test set had 860 tweets, which had 620 not offensive and 280 offensive tweets. Table 1 show the data distribution of the entire dataset. For the current work, train and test data are taken which is 14,100 tweets in number.

Target Class	Train data	Validation data	Test data
NOT	8840	243	620
OFF	4400	77	280

Table 1: Illustrates the Data distribution

In Sub-task A: Offensive language identification, the goal was to discriminate offensive and not-offensive twitter posts. The target classes for each instance were a) Offensive (OFF): posts that contain any form of profanity or targeted offence. This includes threats, insults, and any form of untargeted profanity. b) Not Offensive (NOT): posts that doesn't have any profanity or offense in it. The result and discussion of the top 10 teams for the sub-task A is in section for Introduction. In that, team [4] obtained highest f1-score of 82.9%

4.2. Results and Comparisons

This section describes the result as three different cases. Case 1 & 2 provides baseline approach to compare with the proposed RKS approach described in case 3. Table 2 gives the results of the top 5 teams of sub-task A. Team with rank 1 achieved a maximum f1-score of 82.9%.

Rank	Team Name	System	F1
1.	NULI [4]	Bert	0.829
2.	Nikolov-Radivchev [5]	Bert-Large	0.815
3.	UM-IU@LING [6]	Bert-base-uncased	0.814
4.	Embeddia [7]	Bert	0.808
5.	MIDAS [8]	Bi-LSTM with attention, and Bi-LSTM + Bi-GRU	0.807

Table 2: Illustrates results of top 5 teams in semeval 2019: Task 6 sub-task A

4.2.1. Case 1: Embeddings approach

In this work, we have selected word vectors generated by Google universal encoder model, Fasttext, and DMD based features. The classification using the selected features are performed using machine learning algorithms such as Random Forest (RF), Decision Tree (DT), Naive Bayes (NB), Support vector machine (SVM) linear and RBF kernels, Logistic Regression, and Random kitchen sinks. The evaluation measures used are accuracy (Acc.), precision (Prec), recall, f1-score (F1). Table 3 shows the classification result obtained for classical machine learning algorithms using the Google universal sentence encoder model features. It can be observed that svm linear classifier and Logistic regression has given maximum accuracy of 82.44% and 82.56%.

Google Embedding Vectors				
Algorithm	Acc. (%)	Prec (%)	Recall (%)	F1 (%)
RF (n.estm=30)	77.56	78.89	61.71	62.68
NB	74.19	68.44	69.58	68.92
SVM Linear	82.44	81.13	72.63	75.10
SVM RBF	72.09	36.05	50.00	41.89
LR	82.56	81.71	72.45	75.04

Table 3: Performance evaluation of Universal encoder model features using classical machine learning algorithms

Fasttext Embeddings				
Algorithm	Acc. (%)	Prec. (%)	Recall (%)	F1 (%)
RF (n.estm=30)	76.98	75.84	61.56	62.53
NB	52.33	58.40	59.78	51.89
SVM Linear	81.16	82.99	68.29	70.94
SVM RBF	74.42	79.88	54.68	51.38
LR	81.16	82.99	68.29	70.94

Table 4: Performance evaluation of Fasttext model features using classical machine learning algorithms

Table 4 shows the classification results obtained using the features generated by fasttext model for classical machine learning algorithms. For the fasttext model also, svm linear and logistic regression model have given maximum accuracies of 81.16% respectively.

4.2.2. Case 2: DMD approach

In order to provide a comparison, we explore DMD based features. The Table 5 shows the result obtained

for normal DMD and HODMD based feature. The order for HODMD for the present work is 2 & 3. The classification is performed using SVM-linear kernel with control parameter value chosen as 1000 as the suitable one. We tried for other values such as 0.1, 1, 100, 500, and 1000. Figure 2 shows the control parameter versus accuracy plot which helped to fix the parameter value.

	HODMD		DMD
order → Score ↓	d=2	d=3	
Acc. (%)	77.67	77.33	79.19
Prec. (%)	74.59	74.18	76.63
Recall (%)	64.47	63.72	67.31
F1 (%)	66.14	65.25	69.36

Table 5: Performance evaluation of DMD and HODMD features using SVM linear for control parameter fixed as 1000

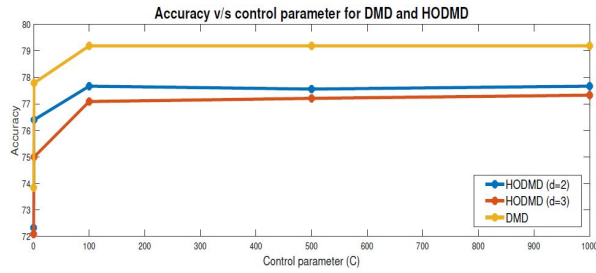


Figure 2: Illustrates the Accuracy v/s control parameter for DMD and HODMD

4.2.3. Case 3: RKS approach

RKS approach has been used in the articles for NLP tasks [29,30,23]. In this work, we use RKS to improve the evaluation scores as discussed previously. The RKS approach explicitly maps the embedding vectors to a dimension where the data becomes linearly separable. In that space, regularized least-square based classification (RLSC) is performed. The implementation of the RKS is taken from [31, 32]. The feature vectors from Google universal sentence encoder and fasttext are explicitly mapped using RKS and the results are tabulated in Table 6 and 7.

Google Embedding - RKS				
Dim → Score ↓	Dim =100	Dim = 200	Dim =500	Dim =1000
Acc.(%)	82.79	84.53	90.47	90.58
Prec.(%)	77.38	81.29	90.31	90.36
Recall (%)	54.17	57.92	73.75	74.17
F1 (%)	63.73	67.64	81.19	81.46

Table 6: Performance evaluation of proposed method using Universal encoder model features

The Table 6 shows the classification report on the proposed RKS method taking word vectors generated by Google universal encoder model as features with dimension 512. For this work, such vector is explicitly mapped to dimensions 100, 200, 500 and 1000 using RKS. The maximum accuracy obtained is 90.58% for higher dimension 1000.

Fasttext Embedding - RKS				
Dim→ Score↓	Dim= 100	Dim= 200	Dim= 500	Dim= 1000
Acc. (%)	85.35	91.05	98.60	99.53
Prec. (%)	85.19	90.95	100.00	99.58
Recall (%)	57.50	75.42	95.00	98.75
F1 (%)	68.66	82.46	97.44	99.16

Table 7: Performance evaluation of proposed method using Fasttext model features

Table 7 shows the classification report on the proposed RKS method taking word vectors generated by Fasttext model as features. For this model also, features are mapped to dimensions 100, 200, 500 and 1000. For Fasttext model, the proposed method gave a maximum accuracy of 99.53%, which is a bench marking result when compared to the literature. This result shows the discriminating capability of the features chosen, as when mapped to higher dimensions, they become linearly separable. From Table 6 and 7 it can be observed that as the mapping dimension increases, the evaluation score also improves. This shows the effectiveness of the RKS approach to obtain competing score. The capability of the RKS approach can be explored on large datasets.

5. Conclusion

Offensive language detection is an important task related to social media data analysis. The nature of the content can vary as its provided by different people. The current work uses the data provided in SemEval 2019 shared task A for Offensive language identification. A comparative study is provided by exploring the effectiveness of Google universal sentence encoder, Fasttext based embedding, Dynamic Mode Decomposition based features and RKS based explicit mapping approach. For the experiments, we used the machine learning methods such as SVM linear, Random Forest, Logistic regression, Navie Bayes and Regularized least-square based classification. The measures used for evaluation are accuracy, precision, recall, and f1-score.

We observed that RKS approach improved the results. However, as a future work, the proposed approach can be explored on large datasets.

References

- [1] Cheng, J., Bernstein, M., Danescu-Niculescu-Mizil, C., & Leskovec, J. (2017, February). Anyone can become a troll: Causes of trolling behavior in online discussions. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing* (pp. 1217-1230). ACM.
- [2] Xu, J. M., Jun, K. S., Zhu, X., & Bellmore, A. (2012, June). Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies* (pp. 656-666). Association for Computational Linguistics.
- [3] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019). Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.
- [4] Liu, P., Li, W., & Zou, L. (2019, June). NULI at SemEval-2019 Task 6: Transfer Learning for Offensive Language Detection using Bidirectional Transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 87-91).
- [5] Nikolov, A., & Radivchev, V. (2019, June). Nikolov-Radivchev at SemEval-2019 Task 6: Offensive Tweet Classification with BERT and Ensembles. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 691-695).
- [6] Zhu, J., Tian, Z., & Kübler, S. (2019). UM-IU@ LING at SemEval-2019 Task 6: Identifying Offensive Tweets Using BERT and SVMs. *arXiv preprint arXiv:1904.03450*.
- [7] Pelicon, A., Martinc, M., & Novak, P. K. (2019, June). Embeddia at SemEval-2019 Task 6: Detecting Hate with Neural Network and Transfer Learning Approaches. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 604-610).
- [8] Mahata, D., Zhang, H., Uppal, K., Kumar, Y., Shah, R., Shahid, S., ... & Anand, S. (2019, June). MIDAS at SemEval-2019 Task 6: Identifying Offensive Posts and Targeted Offense from Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 683-690).
- [9] Schmidt, A., & Wiegand, M. (2017, April). A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media* (pp. 1-10).
- [10] Yin, D., Xue, Z., Hong, L., Davison, B. D., Kontostathis, A., & Edwards, L. (2009). Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2, 1-7.
- [11] Dadvar, M., Trieschnigg, D., Ordelman, R., & de Jong, F. (2013, March). Improving cyberbullying detection with user context. In *European Conference on Information Retrieval* (pp. 693-696). Springer, Berlin, Heidelberg.
- [12] Wiegand, M., Siegel, M., & Ruppenhofer, J. (2018). Overview of the germeval 2018 shared task on the identification of offensive language.
- [13] Kumar, R., Ojha, A. K., Malmasi, S., & Zampieri, M. (2018, August). Benchmarking aggression identification in social media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)* (pp. 1-11).
- [14] <https://fasttext.cc/docs/en/pretrained-vectors.html> (Accessed on July 2019).
- [15] <https://towardsdatascience.com/use-cases-of-googles-universal-sentence-encoder-in-production-dd5aab4fc15>
- [16] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135-146.
- [17] Cer, D., Yang, Y., Kong, S. Y., Hua, N., Limtiaco, N., John, R. S., ... & Sung, Y. H. (2018). Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.
- [18] Loper, E., & Bird, S. (2002). NLTK: the natural language toolkit. *arXiv preprint cs/0205028*.
- [19] Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656, 5-28.
- [20] Grosek, J., & Kutz, J. N. (2014). Dynamic mode decomposition for real-time background/foreground separation in video. *arXiv preprint arXiv:1404.7592*.
- [21] Mohan, N., Soman, K. P., & Kumar, S. S. (2018). A data-driven strategy for short-term electric load forecasting using dynamic mode decomposition model. *Applied energy*, 232, 229-244.
- [22] Sikha, O. K., Kumar, S. S., & Soman, K. P. (2018). Salient region detection and object segmentation in color images using dynamic mode decomposition. *Journal of Computational Science*, 25, 351-366.
- [23] Kumar, S. S., Kumar, M. A., Soman, K. P., & Poornachandran, P. (2020). Dynamic Mode-Based Feature with Random Mapping for Sentiment Analysis. In *Intelligent Systems, Technologies and Applications* (pp. 1-15). Springer, Singapore.
- [24] <https://link.springer.com/book/10.1007/978-981-13-6095-4> (Accessed on July 2019)
- [25] Kayal, S., & Tsatsaronis, G. (2019, July). EigenSent: Spectral sentence embeddings using higher-order Dynamic Mode Decomposition. In *Proceedings of the 57th Conference of the Association for Computational Linguistics* (pp. 4536-4546).
- [26] Rahimi, A., & Recht, B. (2008, September). Uniform approximation of functions with random bases. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing* (pp. 555-561). IEEE.
- [27] Rahimi, A., & Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in neural information processing systems* (pp. 1177-1184).
- [28] Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov), 2579-2605.
- [29] Kumar, S. S., Premjith, B., Kumar, M. A., & Soman, K. P. (2015, December). AMRITA_CEN-NLP@ SAIL2015: sentiment analysis in Indian Language using regularized least square approach with randomized feature learning. In *International Conference on Mining Intelligence and Knowledge Exploration* (pp. 671-683). Springer, Cham.
- [30] S. Thara, A. Krishna, Aspect sentiment identification using random fourier features, in: *International Journal of Intelligent Systems and Applications (IJISA)*, 2018.
- [31] <https://github.com/LCSL/GURLS> (Accessed on July 2019)
- [32] Tacchetti, A., Mallapragada, P. K., Santoro, M., & Rosasco, L. (2013). GURLS: a least squares library for supervised learning. *The Journal of Machine Learning Research*, 14(1), 3201-3205.