CS771 - Intro to ML : Mini Project 1
Group 36 - CerebroX

Ashvani Kumar Yadav
241110013

Awanish Kumar
241110015

Khushwant Kaswan
241110035

Sangharsh Nagdevte
241110064

Souvik Sarkar
231160402

Contents

1 Summary of Validation Accuracies

The best validation set accuracies obtained on the datasets are as follows:

Dataset (100%)	**Validation Accuracy**	**Model Used**
Emoticons as Features Dataset	96.52%	LSTM with 9969 parameters
Deep Features Dataset	98.36%	Logistic Regression
Text Sequence Dataset	88.75%	GRU with 9497 parameters
Combined	98.97%	Logistic Regression

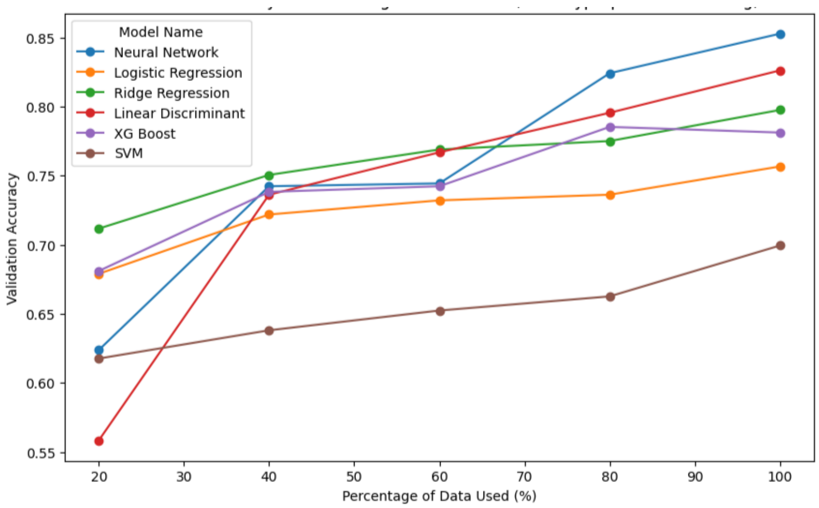
2 Task 1

2.1 Emoticons as Features Dataset

We have tried the following approaches:

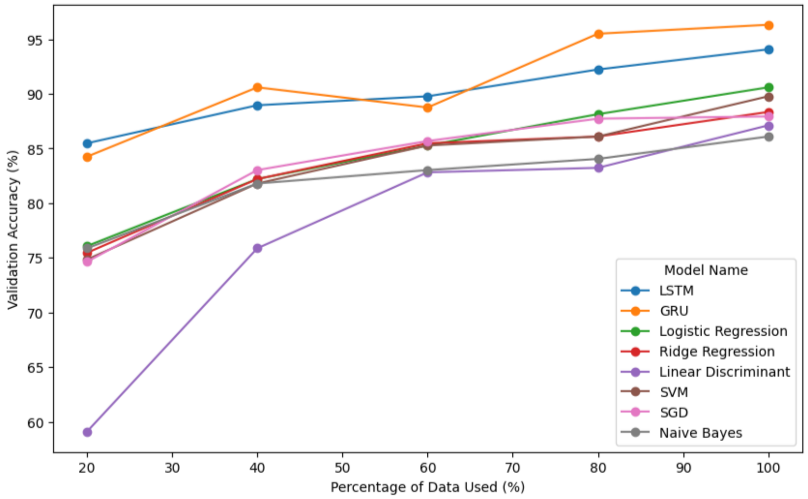
2.1.1 Using pretrained emoji2vec embeddings

Using the pretrained emoji2vec embeddings [1] [2] we converted each emoji to a 300d vector representation converting our training data to shape (7080, 13, 300).Similar transformation on validation and test data. We trained a GRU based model with 9728 parameters resulting in validation set accuracy of **84-86%**. We also trained several other ML models by flattening the dataset to 3900 features. We got max validation set accuracy of about **82.61%** using Linear Discriminant Analysis.



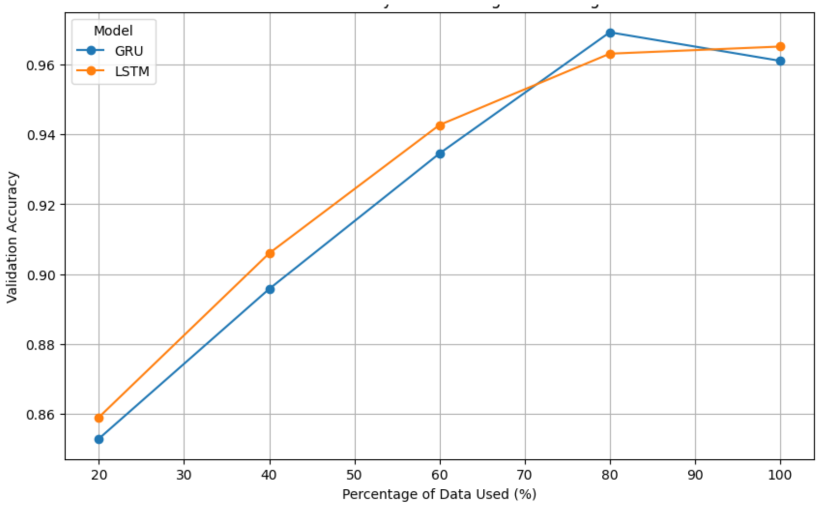
2.1.2 One hot encoding

There were 214 unique emojis in the training dataset. We converted each emoji of all datasets to a 214d one hot vector. We trained a BidirectionalLSTM (9769 parameters) and GRU (9963 parameters) on the modified dataset and got an val-accuracy of **94.07% with LSTM** and **96.32% with GRU**. We also trained sevearal ML models by flattening the dataset. We got max validation set accuracy of about **90.59% with Logistic Regression**.



2.1.3 LSTM and GRU with Embedding Layer

We used an Embedding layer to convert each emoji to a 32d embedding using LSTM model with 9969 parameters and GRU with 9465 parameters. The validation set accuracies obtained with **LSTM is 96.52%** and with **GRU 96.11%**.



We also converted each emoji to their corresponding unicode representation and trained models on it but the resulting accuracy was low compared to above mentioned methods.

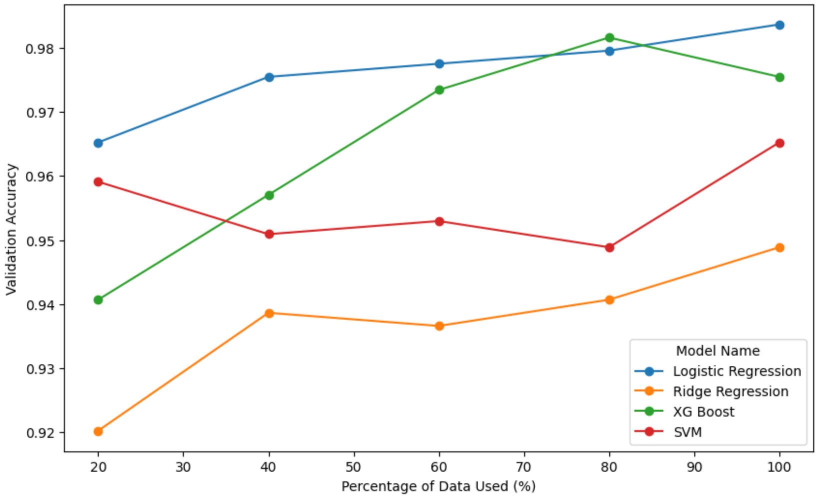
2.2 Deep Features Dataset

2.2.1 Flatten the features

We flattened the training dataset from shape (7080, 13, 768) to shape (7080, 9984). Similar for validation and test data. The max validation set accuracy obtained around 98% using Logistic Regression.

2.2.2 Flatten then Dimensionality Reduction

We flattened the training dataset as above. We reduced the features from 9984 to 500 using PCA retaining 100% variance and getting similar performance of models when compared to original features as above. The max validation set accuracy obtained is **98.36% using Logistic Regression**.



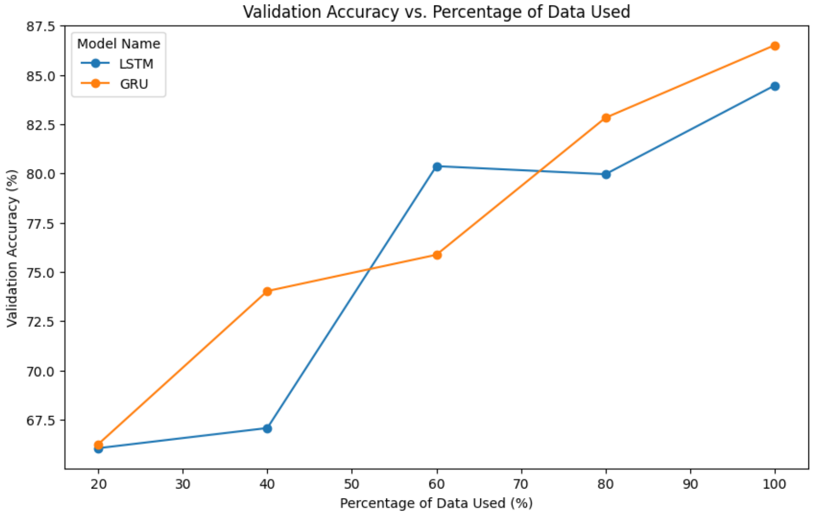
2.3 Text Sequence Dataset

2.3.1 LSTM with 32d embeddings

We used an Embedding layer to convert each digit to a 32d embedding using LSTM model with 9933 parameters. The max validation set accuracies obtained is **84.46%**.

2.3.2 GRU with 64d embeddings

We used an Embedding layer to convert each digit to a 64d embedding uisng GRU model with 9497 parameters. The max validation set accuracies obtained is **86.50%**.



We also trained ML models on the raw dataset by converting into 50 numerical features but the resulting validation accuracy was very low compared to above mentioned methods.

3 Task 2

3.1 M1.1: emoji2vecTransformedEmbedding + 500features + Direct

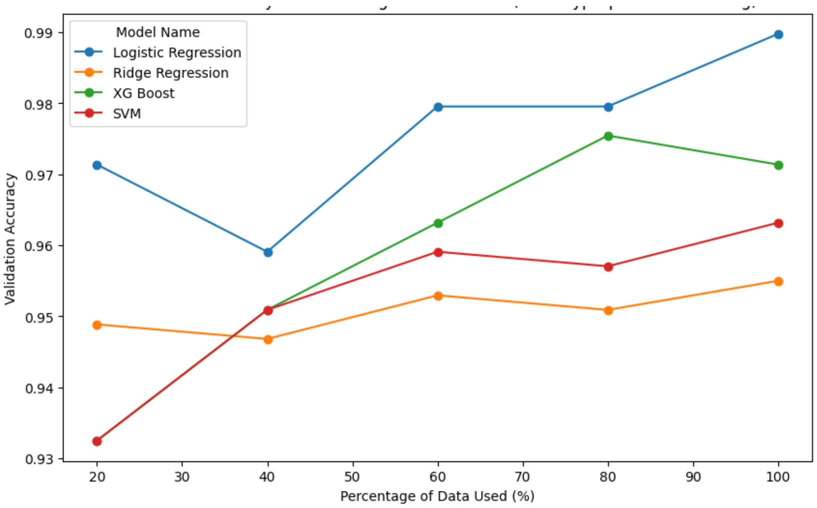
We transformed each emoji of dataset1 to a 300d embedding using the pretrained emoji2vec embeddings resulting in (7080,13,300) shape training data which we later flattened to (7080,3900) and later reduced to (7080,1500) using PCA retaining 100% variance.

The above dataset1 is merged with the reduced 500 feature version of dataset2 resulting in (7080,2000).

We now merge dataset3 directly with the above merged dataset resulting in (7080,2050) training dataset.

Similar transformations are done on the validation and training dataset.

The max validation-set accuracy obtained is **98.97% using Logistic Regression**. **It is marginally greater than the highest accuracy 98.36% obtained in Task1.**



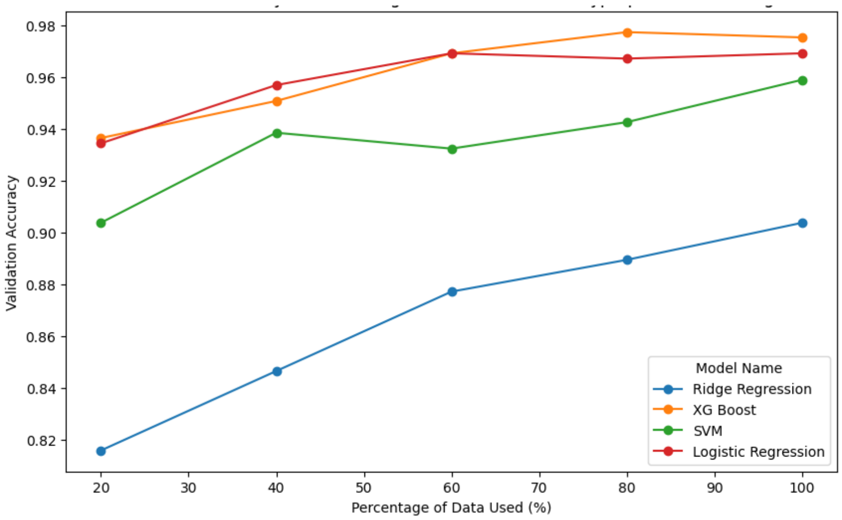
3.2 M1.2: emoji2vecTransformedEmbedding + 500features + TransformedEmbedding

Similar as above we transformed the dataset 1 to shape (7080,1500) and merged with dataset2 same as above resulting on (7080,2000) dataset.

We transformed each number of dataset3 to a 64d embedding that were learnt during the model's training resulting in (7080,50,64) shape training data which we later flattened to (7080,3200).Using PCA reduced features to 450 retaining 100% variance. We now merge it with the above merged dataset resulting in (7080,2450) training dataset.

Similar transformations are done on the validation and training dataset.

The validation-set accuracy obtained is **97.34% using XGBClassifier**. **It is less than the highest accuracy 98.36% obtained in Task1.**



3.3 M2: OneHot + 500features + Direct

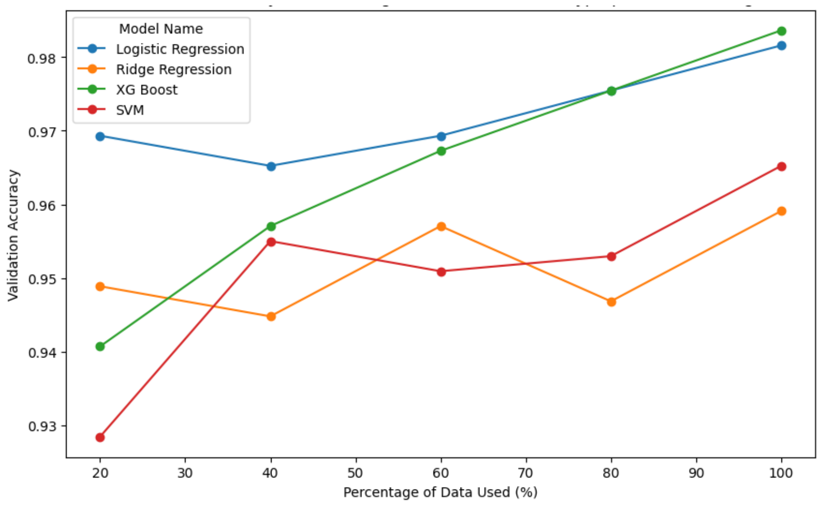
We transformed each emoji of dataset1 to a 214d one hot vector that were learnt during the model's training resulting in (7080,13,214) shape training data which we later flattened to (7080,2782).

The above dataset1 is merged with the reduced 500 feature version of dataset2 resulting in (7080,3282).

We simply merged the 50 features of dataset 3 with the above merged dataset resulting in dataset of shape (7080,3332) .

Similar transformations are done on the validation and training dataset.

The max validation-set accuracy obtained is **98.36% using XGBClassifier**. **It is equivalent to the the highest accuracy 98.36% obtained in Task1.**



3.4 M3: TransformedEmbedding + 500features + TransformedEmbedding

Instead of directly combining the raw features from dataset 1 and 3, we used embedding layers to generate dense vector representations of both the emoticons and the numeric strings.

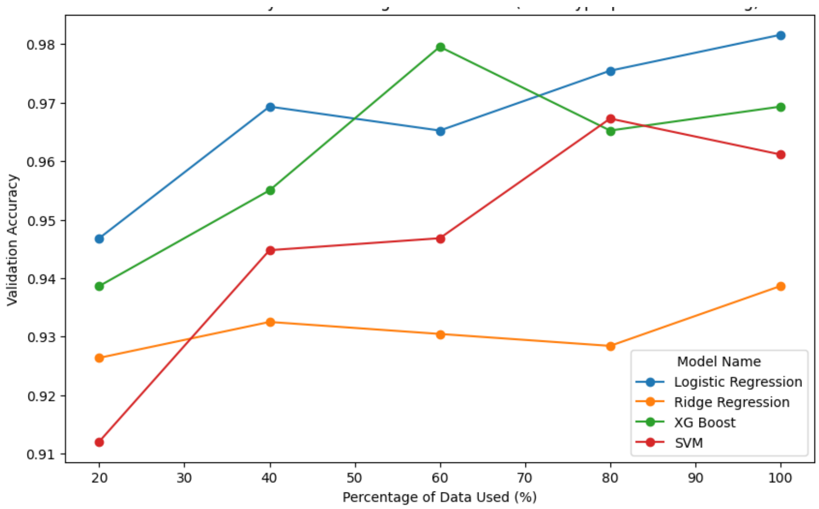
We transformed each emoji of dataset1 to a 32d embedding that were learnt during the model's training resulting in (7080,13,32) shape training data which we later flattened to (7080,416) shape.

The above dataset1 is merged with the reduced 500 feature version of dataset2 resulting in dataset of shape (7080,916).

We transformed each number of dataset3 to a 64d embedding that were learnt during the model's training resulting in (7080,50,64) shape training data which we later flattened to (7080,3200).Using PCA reduced features to 450 retaining 100% variance. We now merge it with the above merged dataset resulting in (7080,1366) training dataset.

Similar transformations are done on the validation and training dataset.

The validation-set accuracy obtained is **98.15% with Logistic Regression**. **It is marginally less than the highest accuracy 98.36% obtained in Task1.**



4 Possible Caveats

1. Validation Set accuracies in the LSTM/GRU models may change/fluctuate a little with each run as we have used Dropout layers which contribute to some variability.
2. Accuracies for Task2 M3 depends somewhat on how well the previous models have performed on dataset1 and dataset3 providing us with good embeddings for transformation.
3. For emojis that were not availabe in emoji2vec embeddings we have used a 300d zero vector.
4. For emojis that were not present in training data but present in test we have used a zero vector in case of OneHotEncoding too.
5. Hyperparameter Tuning was done without cross validation. It was done using a custom_scorer based on parameter's accuracy on the validation dataset.

```
grid_search = GridSearchCV(
estimator=model,
param_grid=param_grid,
scoring=custom_scorer,
refit=False,
cv=[(slice(None), slice(None))]
# No real cross-validation
)
grid_search.fit(X_subset, y_subset)
```

Acknowledgments

We would like to express our sincere gratitude to our course instructor, **Dr. Piyush Rai**, for their invaluable guidance throughout the course of this project. Their feedback were crucial in shaping the direction of our work.

We would also like to extend our appreciation to our team members. Each member played an integral role in the successful completion of this project.

Additionally, we are grateful for the online resources, including scikit-learn documentation and Stack Overflow , which provided helpful solutions to some of the challenges we faced. The tools and libraries, including TensorFlow, Keras, and Scikit-learn, were instrumental in the completion of our mini machine learning project.

References

[1] Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. emoji2vec: Learning emoji representations from their description. In Lun-Wei Ku, Jane Yung-jen Hsu, and Cheng-Te Li, editors, *Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media*, pages 48–54, Austin, TX, USA, November 2016. Association for Computational Linguistics.

[2] https://github.com/uclnlp/emoji2vec.