CS771 - Intro to ML: Mini-project 1

Group Members

Kirandeep Parida
241110036

Arnika Kaithwas
241110011

Prabhakar Pandey
241110049

Tsewang Chukey
241110092

Tsewang Namgail
241110093

Department of Computer Science and Engineering

IIT Kanpur

October 2024

1 Abstract

This report investigates the development of binary classification models using three datasets with different feature encodings: Emoticon-based features, Deep neural network-derived features, and Text sequences. The goal is to build models with high accuracy while minimizing training data. The first task identifies the best model for each dataset by varying training sizes and evaluating validation accuracy. The second task explores whether combining all datasets improves performance. The report outlines the models, feature transformations, and experimental results, comparing the strengths and limitations of individual and merged datasets. Predictions on test sets are made using the top-performing models to analyze the impact of feature representation on performance.

2 Introduction

The primary aim of this project is to develop the best binary classification model for each dataset while minimizing the amount of training data used, thereby optimizing both performance and efficiency. Additionally, we explore whether combining the datasets yields better results compared to using them independently. The report is structured to present the experimental approach, results, and insights from individual models, followed by the findings from combining the datasets. The performance of each model is evaluated based on accuracy on the validation/test set and the ability to generalize using minimal training data.

3 Process Steps

1. Loading the dataset
2. Preprocessing the dataset
3. Build the model
4. Train the model
5. Evaluate the model on validation set
6. Generate the predictions on test set

4 Analysis and Results

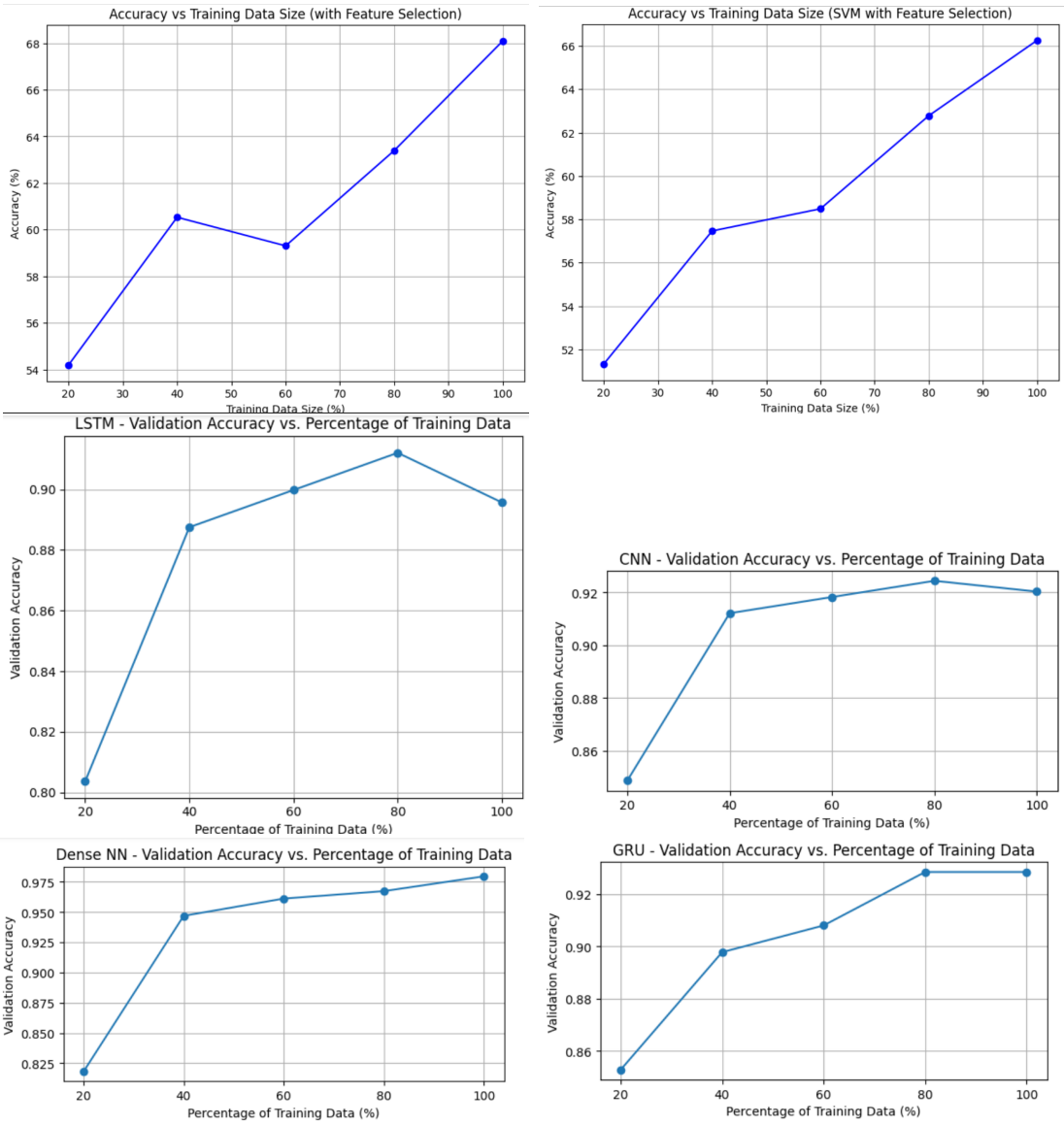
4.1 Task 1: Model Creation on Individual Dataset

4.1.1 Emoticons as Features Dataset

On this dataset we have tried multiple models i.e. Logistic Regression, SVM, CNN, LSTM, DNN and GRU and DNN is giving the best accuracy on all proportion of training example. So, we are choosing DNN for the test set prediction.

% of training example	Logi. Reg.	SVM	CNN	DNN	LSTM
20%	54.19	51.33	84.87	81.80	80.37
40%	60.53	57.46	91.21	94.68	88.75
60%	59.30	58.49	91.82	96.11	89.98
80%	63.39	62.78	92.43	96.73	91.21
100%	68.10	66.26	92.02	97.96	89.57

Table 1: Validation accuracy of Models used corresponding to different proportions of training example

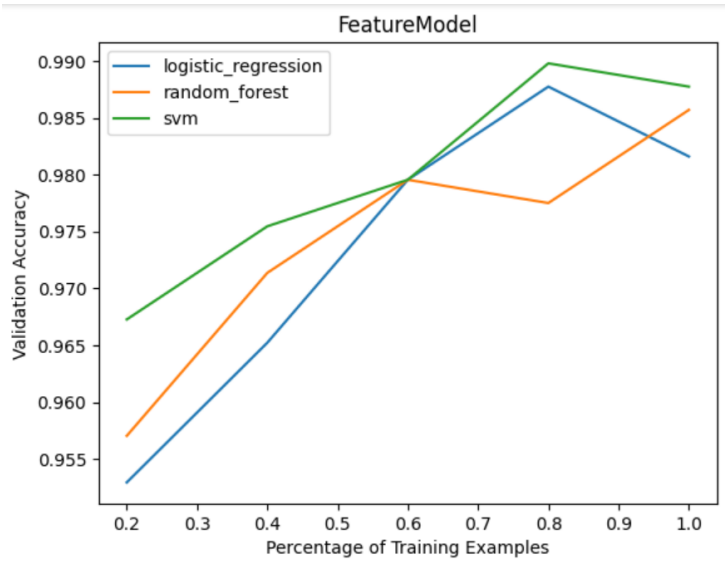


4.1.2 Deep Features Dataset

On this dataset we have tried three models i.e. Logistic Regression, SVM and Random forest and SVM is giving better accuracy on every proportion of training example as compare to other two models used. So we are choosing SVM for test set prediction among all the models we have tried.

% of training example	Logistic Regression	SVM	Random Forest
20%	95.29	96.72	95.70
40%	96.52	97.54	97.13
60%	97.95	97.95	97.95
80%	98.77	97.75	97.75
100%	98.15	98.56	98.56

Table 2: Validation accuracy of Models used on different proportions of training example.



4.1.3 Text Sequence Dataset

For this dataset we have tried two models SVM and CNN and CNN is giving the better accuracy on the validation set but the trainable parameters are very high in this model. So, for the test set prediction we are choosing SVM over CNN.

% of training example	SVM	CNN
20%	60.94	72.59
40%	64.01	89.34
60%	66.05	82.00
80%	66.26	80.57
100%	65.44	85.27

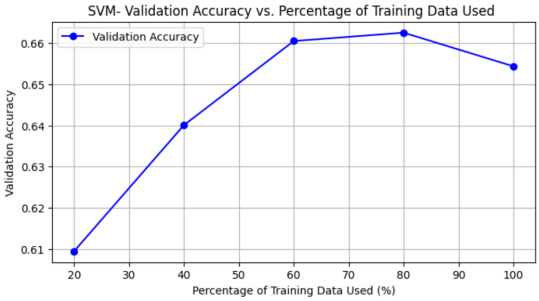
Table 3: Validation accuracy of Models used

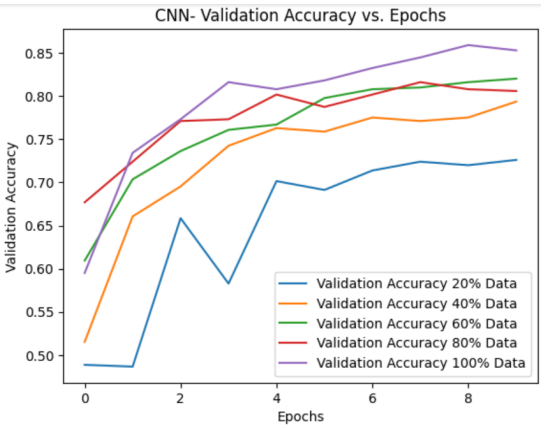
Below the plots are showing validation set accuracy corresponding to different proportion of the training example.

4.2 Task 2: Combining all three Dataset

4.2.1 Emoticon Dataset

- **Preprocessing:** Emoticon sequences were tokenized using a character-level tokenizer. The tokenized sequences were padded to ensure uniform length across the data.





- **Model Architecture:** An embedding layer is used to convert the tokenized input into vectorized form.
- A Bidirectional LSTM layer processes the sequence data, followed by a Dropout layer to prevent overfitting.

4.2.2 Deep Feature Dataset

- **Preprocessing:** This dataset contains features with a shape of (N, 13, 786). The 13 feature vectors were averaged to reduce dimensionality, resulting in (N, 786) vectors.
- **Model Architecture:**A dense layer with ReLU activation was applied to the processed deep features, followed by Dropout to avoid overfitting.

4.2.3 Text Sequence Dataset

- **Preprocessing:** The input strings were tokenized using the same character-level tokenizer as the emoticon dataset. The tokenized sequences were padded to a uniform length.
- **Model Architecture:** An embedding layer was used, followed by a Bidirectional LSTM to capture sequential dependencies in the text, and a Dropout layer for regularization.

4.2.4 Combined Model

- The three branches (emoticon, deep features, and text) were concatenated into a single feature vector.
- Fully connected layers were used after concatenation, with ReLU activation and Dropout for final classification.
- The output layer used a sigmoid activation function for binary classification(1 or 0).

The total trainable parameters used : **9,555**
Model Validation Accuracy : **95.50**

The use of Bidirectional LSTMs helps in capturing the contextual information in both emoticons and text sequences. The model architecture, including dropout layers, effectively mitigates overfitting.

Further improvements can be made by experimenting with different hyperparameters, increasing training data, or using more sophisticated feature extraction techniques for text sequences.

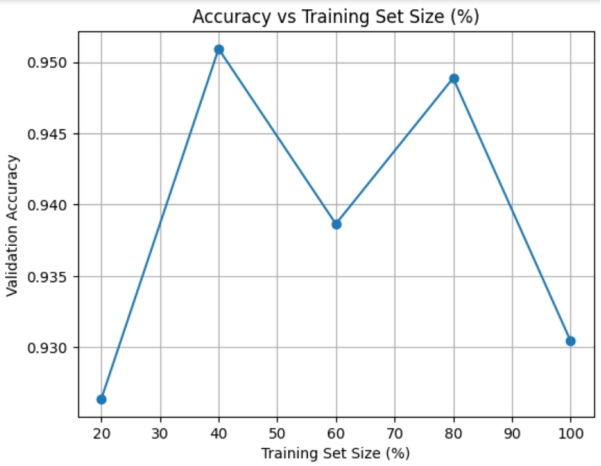


Figure 1: Validation Accuracy corresponding to different proportion of training example.

4.2.5 Rationale for Trying Different Models on the Datasets

We have tried different models on each dataset due to following reasons:

- **Distinct Feature format:** The three datasets provided represent the same data through different features: emoticons, deep features, and text sequences. These representations differ in complexity, structure, and dimensionality, prompting us to experiment with various models to determine which best accommodates the unique characteristics of each dataset.

- **Model Performance and Generalization:** By testing a variety of models (e.g., logistic regression, decision trees, or neural networks), we have compared their performance across metrics like accuracy, precision. Since, our aim to find a model that not only achieves high accuracy but also generalizes well across different datasets.

- **Optimizing Accuracy and Minimal Trainig Dataset:** Some models perform well with limited data, others need more for optimal results. We have tested various models so that we can find those which achieve high accuracy with less data.

- **Combining Insights Across Datasets:**In addition to testing models on each dataset, we have explored whether combining insights from all three can create a stronger model. This approach leverages the unique strengths of each dataset to potentially outperform individual models.

5 Conclusion

In this mini-project, we explored three binary classification datasets, each with unique feature representations derived from the same raw data. For each dataset, we trained multiple machine learning models and identified the best-performing models based on validation accuracy and efficiency in using minimal training data. We further investigated whether combining the datasets, which provided complementary information through diverse feature sets, could enhance model performance.

This project provided valuable insights into the importance of feature representation and the trade-offs between training data size and model generalization. The results emphasize that both dataset-specific models and combined approaches have merits depending on the nature of the problem and features used.

6 Acknowledgement

We are deeply grateful to **Prof. Piyush Rai**, the instructor for CS771 - Introduction to Machine Learning, for providing us with the opportunity to work on this mini-project. Their expert guidance and constructive feedback played a crucial role in shaping our understanding and approach to this project.

A special thanks to our group members, whose dedication and collaboration were vital to the successful completion of this project. Working together as a team has greatly enriched our learning experience and has been key to overcoming various challenges. Lastly, we would like to acknowledge the use of open-source libraries such as Scikit-learn, which provided essential tools and resources that allowed us to experiment with and implement machine learning models effectively.

References

[1] Chollet, F., & others. (2015). *Keras: Deep Learning for Humans*. `https://keras.io`

[2] Abadi, M., Agarwal, A., Barham, P., & et al. (2016). *TensorFlow: Large-scale Machine Learning on Heterogeneous Systems*. arXiv preprint arXiv:1603.04467. `https://arxiv.org/abs/1603.04467`

[3] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[4] TensorFlow Hub: Google's NNLM (Neural Network Language Model) embeddings. `https://tfhub.dev/google/nnlm-en-dim50/2`.

[5] Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*. `https://arxiv.org/abs/1408.5882`.