WildChat Conversation Analysis: Language Identification and Conversation-Length Prediction

Harish Maheswaran
Indiana University Bloomington
hamahes@iu.edu

Abstract

This project analyzes the WildChat dataset to study how users interact with ChatGPT-like systems across languages and conversation styles. We focus on two research questions. First, we evaluate whether a lightweight text-classification model can accurately predict the language of user prompts across six languages (de, en, es, fr, it, pt). Second, we investigate whether conversation-level features can predict whether an interaction is short or long. Using standard preprocessing and TF–IDF features, the language identification model achieves 0.913 accuracy on a held-out test set with strong performance on English and moderate degradation on minority-language classes. For conversation-length prediction, the baseline model achieves 0.609 accuracy but shows severe class imbalance effects, with near-zero recall for the long class. These findings highlight both the strength of simple models for language detection and the need for better balancing strategies and richer features when modeling interaction length.

Code — https://github.com/Harish-m-07/Assignment-1-Text-Analysis

Introduction

Large-scale conversational datasets make it possible to study real user interactions with modern language models. Understanding language usage and conversation length is useful for product analytics, personalization, and evaluation of user engagement. From a data-science perspective, language identification is a foundational step for multilingual analytics pipelines, while conversation-length modeling provides signals about user effort, satisfaction, and complexity of requests.

In this work, we use the WildChat dataset provided for the course to explore two practical modeling tasks: (1) predicting the language of a prompt, and (2) predicting whether an interaction is short or long. We deliberately start with simple and reproducible baselines (TF–IDF + classical classifiers) because they are interpretable, quick to train, and often strong in text classification (Salton and Buckley 1988; McCallum and Nigam 1998; Pedregosa et al. 2011; Joachims 1998; Sokolova and Lapalme 2009).

Related Work

Traditional text classification pipelines commonly use bag-of-words or TF–IDF representations combined with linear models or probabilistic classifiers (Salton and Buckley 1988; McCallum and Nigam 1998; Joachims 1998; Sokolova and Lapalme 2009). For language identification, character- and word-based n-gram approaches have historically performed well, and modern systems often incorporate neural encoders; however, TF–IDF baselines remain competitive for coarse language categories when the label set is small and the texts are short.

Conversation modeling has been studied using dialogue act tagging, turn-level prediction, and engagement/retention signals. Predicting conversation length is more challenging because length depends on many latent factors (task complexity, user persistence, model helpfulness) and commonly suffers from class imbalance if long conversations are relatively rare.

Dataset and Data Processing

Dataset

WildChat is a dataset of conversational interactions that includes user prompts and assistant responses. For this project, we use two derived supervised-learning datasets:

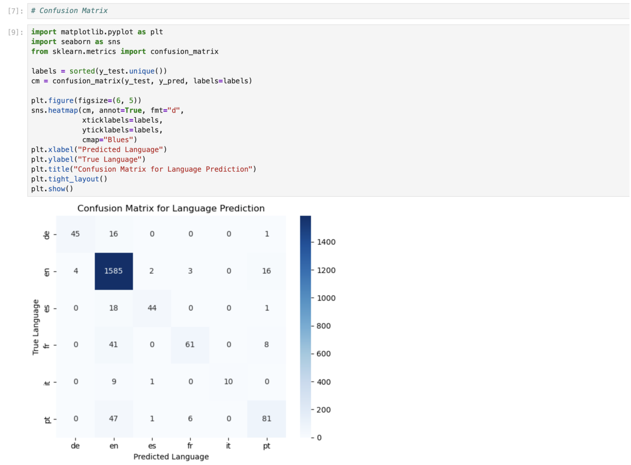
- **Language dataset:** each sample is a user prompt labeled with one of six languages (de, en, es, fr, it, pt).
- **Conversation-length dataset:** each sample is labeled as `Short` or `Long` based on the number of turns or a course-provided rule.

Preprocessing

We apply standard cleaning steps to make modeling consistent and reproducible:

- Remove null/empty texts and normalize whitespace.
- Lowercasing (for word-level TF–IDF).
- Train/test split with a fixed random seed.
- Vectorization using TF–IDF with a capped vocabulary size (to control sparsity).

All experiments were implemented in Python using common ML libraries (Pedregosa et al. 2011).



```
[7]: # Confusion Matrix

[9]: import matplotlib.pyplot as plt
     import seaborn as sns
     from sklearn.metrics import confusion_matrix

     labels = sorted(y_test.unique())
     cm = confusion_matrix(y_test, y_pred, labels=labels)

     plt.figure(figsize=(6, 5))
     sns.heatmap(cm, annot=True, fmt="d",
                 xticklabels=labels,
                 yticklabels=labels,
                 cmap="Blues")
     plt.xlabel("Predicted Language")
     plt.ylabel("True Language")
     plt.title("Confusion Matrix for Language Prediction")
     plt.tight_layout()
     plt.show()
```

Confusion Matrix for Language Prediction

	de	en	es	fr	it	pt
de	45	16	0	0	0	1
en	4	1585	2	3	0	16
es	0	18	44	0	0	1
fr	0	41	0	61	0	8
it	0	9	1	0	10	0
pt	0	47	1	6	0	81

Figure 1: Confusion matrix for language prediction across six languages (de, en, es, fr, it, pt).

Research Question 1: Language Identification (Method, Findings)

Research Question

RQ1: Can a lightweight TF–IDF based classifier accurately predict the language of user prompts in WildChat?

Method

We model language identification as a multi-class text classification problem with six labels. The pipeline is:

1. Convert prompts to TF–IDF vectors.
2. Train a supervised classifier (baseline).
3. Evaluate using accuracy and per-class precision/recall/F1.

We report a classification report and a confusion matrix to understand common confusions between languages.

Findings

The language classifier achieves **0.913 accuracy** on the test set. Performance is strong for English, while minority-language classes show lower recall, suggesting class imbalance and limited examples affect generalization. The confusion matrix indicates many errors map minority languages into English, consistent with a dominant-class bias.

Quantitative Results

Table 1 summarizes the key metrics from the held-out evaluation (from the notebook output).

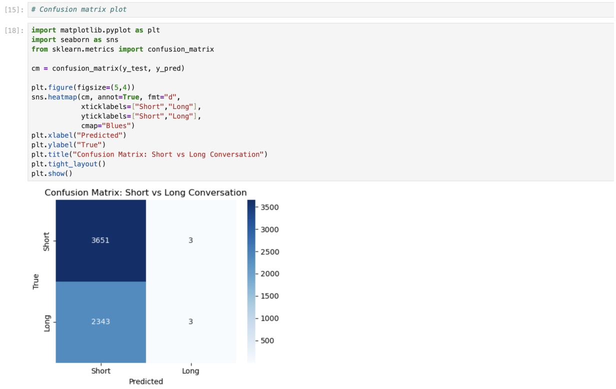
Research Question 2: Conversation-Length Prediction (Method, Findings)

Research Question

RQ2: Can a baseline classifier predict whether a WildChat interaction is `Short` or `Long`?

Lang	**Precision**	**Recall**	**F1**	**Support**
de	0.92	0.73	0.81	62
en	0.92	0.98	0.95	1610
es	0.92	0.70	0.79	63
fr	0.87	0.55	0.68	110
it	1.00	0.50	0.67	20
pt	0.76	0.60	0.67	135
Accuracy			0.913 (N=2000)	

Table 1: RQ1 evaluation metrics from the classification report (language identification).



```
[15]: # Confusion matrix plot

[18]: import matplotlib.pyplot as plt
      import seaborn as sns
      from sklearn.metrics import confusion_matrix

      cm = confusion_matrix(y_test, y_pred)

      plt.figure(figsize=(5,4))
      sns.heatmap(cm, annot=True, fmt="d",
                  xticklabels=["Short","Long"],
                  yticklabels=["Short","Long"],
                  cmap="Blues")
      plt.xlabel("Predicted")
      plt.ylabel("True")
      plt.title("Confusion Matrix: Short vs Long Conversation")
      plt.tight_layout()
      plt.show()
```

Confusion Matrix: Short vs Long Conversation

	Short	Long
Short	3651	3
Long	2343	3

Figure 2: Confusion matrix for conversation-length prediction (`Short` vs `Long`).

Method

We frame conversation-length prediction as binary classification. We use the same general approach:

1. Represent text (or conversation summary fields) with TF–IDF features.
2. Train a baseline classifier.
3. Evaluate using accuracy and per-class metrics plus confusion matrix.

Because conversation-length labels can be imbalanced, we focus on recall and F1 for the minority class.

Findings

The baseline model reaches **0.609 accuracy**, but the detailed metrics show a major issue: the classifier predicts almost everything as `Short`, yielding **near-zero recall for the `Long` class**. This indicates strong class imbalance and suggests that accuracy alone is misleading for this task. The confusion matrix confirms that most `Long` samples are classified as `Short`.

Discussion

This project reveals a clear contrast between the two tasks. Language identification performs well even with a simple baseline because the label space is small and language cues are strongly expressed in surface vocabulary. However, performance drops for languages with fewer samples (e.g., Italian), and the confusion matrix suggests that the majority

Class	**Precision**	**Recall**	**F1**	**Support**
Short	0.61	1.00	0.76	3654
Long	0.50	0.00	0.00	2346
Accuracy			0.609 (N=6000)	

Table 2: RQ2 evaluation metrics from the classification report (conversation length).

class (English) absorbs many mistakes. A straightforward improvement would be class-balanced training, more data for minority languages, or character n-gram features that are robust to short texts.

Conversation-length prediction is substantially harder. The baseline accuracy (0.609) hides the fact that the model fails to detect `Long` conversations (recall ≈ 0). This outcome is consistent with an imbalanced label distribution and feature insufficiency: TF–IDF on raw text may not capture the interaction dynamics that cause long conversations. Practical improvements include using class weights, resampling, threshold tuning, and adding features such as prompt length, number of turns, or semantic embeddings. Reporting macro-F1 and per-class recall is essential for honest evaluation in this setting.

Conclusion

We studied WildChat interactions through two supervised tasks. For language identification, a TF–IDF baseline achieved 0.913 test accuracy, showing that lightweight methods can robustly support multilingual analytics. For conversation-length prediction, the baseline model produced 0.609 accuracy but failed to identify long conversations due to imbalance and limited features. Future work should apply balancing strategies and incorporate interaction-level features to better model engagement and conversation depth.

Appendix: Implementation and Reproducibility

Code and Resources

All code, notebooks, and outputs are available at:

> https://github.com/Harish-m-07/Assignment-1-Text-Analysis

Appendix A: Notes on Evaluation

For RQ1 (language identification), accuracy is informative due to the strong performance on the dominant class, but per-class recall is needed to diagnose minority-language weaknesses. For RQ2, accuracy is not sufficient; macro-F1 and long-class recall are the key indicators because the model can obtain decent accuracy by predicting only the majority class.

Appendix B: Figures Used in This Report

Place the following files in the same directory as `main.tex`:

- `fig_lang_confusion.png` (language confusion matrix screenshot)
- `fig_len_confusion.png` (short vs long confusion matrix screenshot)

Appendix C: Optional Extra Table (Aggregated Scores)

If you want, you can include macro/weighted averages for RQ1 and RQ2 here to provide a complete metrics summary.

References

Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. *European Conference on Machine Learning*, 137–142.

McCallum, A.; and Nigam, K. 1998. A comparison of event models for naive Bayes text classification. In *AAAI Workshop on Learning for Text Categorization*.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Salton, G.; and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5): 513–523.

Sokolova, M.; and Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4): 427–437.