Reproducing User Intent Prediction in Information-seeking Conversations

Project of MU4IN013 RITAL

Shen Du, Xiaofeng Shentu, Vladimir Kazarin

May 2025

1 Introduction

This report aims to *reproduce and evaluate* the models proposed in the paper *User Intent Prediction in Information-seeking Conversations* [1]. The core task is to perform multi-class classification of user utterances within a conversational context.

Following the original paper, our project is structured into two main parts. The first part uses handcrafted features provided in the paper to train traditional machine learning classifiers such as Random Forests and Support Vector Machines (SVM). The second part applies neural architectures, including Convolutional Neural Networks (CNN), to automatically learn representations and perform classification.

Based on the referenced work, our goal is to categorize each utterance from a technical support platform—where users interact with support agents—into one of several fine-grained intent classes. These classes, such as *Original Question*, *Clarifying Question*, or *Positive Feedback*, reflect the role of the utterance in multi-turn information-seeking conversations. An overview of these intent categories is shown in Table 1.

Table 1: User intent taxonomy and distribution in MSDialog

Code	Label	Description	%
OQ	Original Question	The first question from the user to initiate the dialog.	13
RQ	Repeat Question	Other users repeat a previous question.	3
CQ	Clarifying Question	User or agent asks for clarifications.	4
FD	Further Details	User or agent provides more details.	14
FQ	Follow Up Question	User asks for follow up questions about relevant issues.	5
IR	Information Request	Agent asks for information from users.	6
PA	Potential Answer	A potential answer or solution provided by agents.	22
PF	Positive Feedback	User provides positive feedback for working solutions.	6
NF	Negative Feedback	User provides negative feedback for useless solutions.	4
GG	Greetings/Gratitude	Greetings or expressing gratitude.	22
JK	Junk	No useful information in the utterance.	1
O	Others	Utterances cannot be categorized using other classes.	1

2 Traditional Machine Learning Models Reproduction

In the first part of our experiments, we follow the approach described in [1] to extract a set of features from each utterance, constructing a structured feature matrix as illustrated in Table 2. These features span across three categories: *Content*, *Structural*, and *Sentiment*, and are designed to capture various lexical, positional, and emotional properties of each utterance.

2.1 Traditional Classifier Evaluation

We implemented and evaluated five classifiers: Random Forest, Support Vector Machine (SVM), AdaBoost, Naive Bayes, and ML-kNN, using the handcrafted features described in Table 2. In this section, we present both per-class classification reports and overall performance metrics, followed by a comparison with the baseline results reported in [1].

We applied five machine learning algorithms for multi-class classification on the extracted feature vectors: Random Forest, Support Vector Machine (SVM), Multi-Label K-Nearest Neighbor (ML-KNN), Naive Bayes, and AdaBoost. The models were evaluated on both validation and test sets. In the following, we report detailed classification results per class, as well as micro/macro/weighted averages.

Before presenting the classification results of each traditional machine learning model, we briefly explain how to read the classification reports.

Each report is divided into two parts: one for validation data and the other for test data. For each intent class (e.g., `CQ`, `FD`, `PA`), the metrics **precision**, **recall**, and **F1-score** are reported, along with the total number of samples (**support**) belonging to that class.

Table 2: Features extracted for user intent prediction in information-seeking conversations.

Feature Name	**Group**	**Description**	**Type**
Initial Utterance Similarity	Content	Cosine similarity between the utterance and the first utterance of the dialog	Real
Dialog Similarity	Content	Cosine similarity between the utterance and the entire dialog	Real
Question Mark	Content	Does the utterance contain a *question mark*	Binary
Duplicate	Content	Does the utterance contain the keywords *same, similar*	Binary
5W1H	Content	Does the utterance contain *what, where, when, why, who, how*	One-hot vector
Absolute Position	Structural	Absolute position of an utterance in the dialog	Numerical
Normalized Position	Structural	Normalized position (AbsPos divided by total utterances)	Real
Utterance Length	Structural	Total number of words after stopword removal	Numerical
Utterance Length Unique	Structural	Unique words count after stopword removal	Numerical
Utterance Length Stemmed Unique	Structural	Unique stemmed words count	Numerical
Is Starter	Structural	Is the utterance made by the dialog starter	Binary
Thank	Sentiment	Does the utterance contain *thank*	Binary
Exclamation Mark	Sentiment	Does the utterance contain *!*	Binary
Feedback	Sentiment	Contains phrases like *did not, does not*	Binary
Sentiment Scores	Sentiment	Sentiment scores computed by VADER	Real
Opinion Lexicon	Sentiment	Number of positive/negative words from a lexicon	Numerical

- **Precision** indicates how many of the predicted samples for a class are actually correct.
- **Recall** measures how many of the actual samples of a class were correctly identified.
- **F1-score** is the harmonic mean of precision and recall, providing a balanced evaluation.

At the bottom of the table, we include:

- **Micro average**: metrics calculated globally by counting the total true positives, false negatives, and false positives.
- **Macro average**: unweighted mean across all classes (treats each class equally).
- **Weighted average**: mean weighted by support (takes class imbalance into account).
- **Samples average**: applicable for multi-label settings, averaged over samples; included here for completeness.

Understanding these metrics is crucial for evaluating model performance not only in aggregate, but also on minority and majority classes.

2.1.1 Random Forest Results

Validation Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	66
FD	0.58	0.35	0.44	193
FQ	0.36	0.08	0.13	65
GG	0.63	0.46	0.54	41
IR	0.67	0.24	0.35	84
JK	1.00	0.08	0.15	12
NF	0.50	0.02	0.04	47
O	0.00	0.00	0.00	2
OQ	0.98	0.97	0.97	221
PA	0.82	0.86	0.84	365
PF	0.71	0.41	0.52	97
RQ	0.29	0.04	0.07	49
micro avg	0.80	0.55	0.65	1242
macro avg	0.54	0.29	0.34	1242
weighted avg	0.69	0.55	0.58	1242
samples avg	0.68	0.61	0.63	1242

Test Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	46
FD	0.76	0.38	0.51	226
FQ	0.61	0.19	0.29	59
GG	0.64	0.36	0.46	25
IR	0.62	0.23	0.34	103
JK	0.00	0.00	0.00	12
NF	0.00	0.00	0.00	54
O	0.00	0.00	0.00	4
OQ	0.98	0.95	0.96	226
PA	0.81	0.87	0.84	364
PF	0.81	0.40	0.54	87
RQ	1.00	0.12	0.21	51
micro avg	0.83	0.56	0.67	1257
macro avg	0.52	0.29	0.35	1257
weighted avg	0.73	0.56	0.60	1257
samples avg	0.70	0.61	0.64	1257

2.1.2 AdaBoost Results

Validation Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	66
FD	0.53	0.34	0.41	193
FQ	0.50	0.12	0.20	65
GG	0.71	0.49	0.58	41
IR	0.54	0.15	0.24	84
JK	0.33	0.08	0.13	12
NF	0.27	0.06	0.10	47
O	0.00	0.00	0.00	2
OQ	0.98	0.97	0.97	221
PA	0.79	0.86	0.82	365
PF	0.69	0.49	0.57	97
RQ	0.57	0.08	0.14	49
micro avg	0.76	0.56	0.64	1242
macro avg	0.49	0.30	0.35	1242
weighted avg	0.66	0.56	0.58	1242
samples avg	0.67	0.62	0.63	1242

Test Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	46
FD	0.73	0.41	0.53	226
FQ	0.71	0.20	0.32	59
GG	0.65	0.52	0.58	25
IR	0.60	0.17	0.27	103
JK	0.25	0.08	0.12	12
NF	0.50	0.04	0.07	54
O	0.00	0.00	0.00	4
OQ	0.98	0.95	0.96	226
PA	0.81	0.88	0.84	364
PF	0.78	0.37	0.50	87
RQ	0.64	0.14	0.23	51
micro avg	0.82	0.57	0.67	1257
macro avg	0.55	0.31	0.37	1257
weighted avg	0.74	0.57	0.61	1257
samples avg	0.70	0.62	0.64	1257

2.1.3 SVM Results

Validation Classification Report:

	precision	recall	f1-score	support

CQ	0.00	0.00	0.00	66
FD	0.56	0.31	0.39	193
FQ	0.00	0.00	0.00	65
GG	0.58	0.27	0.37	41
IR	0.43	0.04	0.07	84
JK	0.00	0.00	0.00	12
NF	1.00	0.02	0.04	47
O	0.00	0.00	0.00	2
OQ	0.97	0.97	0.97	221
PA	0.76	0.87	0.81	365
PF	0.82	0.46	0.59	97
RQ	0.00	0.00	0.00	49
micro avg	0.79	0.52	0.63	1242
macro avg	0.43	0.24	0.27	1242
weighted avg	0.63	0.52	0.54	1242
samples avg	0.65	0.58	0.60	1242

Test Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	46
FD	0.70	0.31	0.43	226
FQ	0.00	0.00	0.00	59
GG	0.67	0.24	0.35	25
IR	0.62	0.05	0.09	103
JK	0.00	0.00	0.00	12
NF	0.00	0.00	0.00	54
O	0.00	0.00	0.00	4
OQ	0.97	0.95	0.96	226
PA	0.77	0.87	0.82	364
PF	0.87	0.38	0.53	87
RQ	0.00	0.00	0.00	51
micro avg	0.82	0.51	0.63	1257
macro avg	0.38	0.23	0.26	1257
weighted avg	0.65	0.51	0.54	1257
samples avg	0.65	0.57	0.59	1257

2.1.4 Naive Bayes Results

Validation Classification Report:

	precision	recall	f1-score	support

CQ	0.22	0.33	0.27	66
FD	0.41	0.32	0.36	193
FQ	0.26	0.28	0.27	65
GG	0.15	0.95	0.25	41
IR	0.21	0.74	0.32	84
JK	0.03	1.00	0.05	12
NF	0.16	0.32	0.21	47
O	0.01	1.00	0.01	2
OQ	0.93	0.97	0.95	221
PA	0.69	0.87	0.77	365
PF	0.32	0.81	0.46	97
RQ	0.24	0.59	0.35	49
micro avg	0.31	0.70	0.43	1242
macro avg	0.30	0.68	0.36	1242
weighted avg	0.52	0.70	0.57	1242
samples avg	0.43	0.74	0.50	1242

Test Classification Report:

	precision	recall	f1-score	support
CQ	0.20	0.50	0.29	46
FD	0.38	0.30	0.34	226
FQ	0.23	0.32	0.27	59
GG	0.10	0.96	0.19	25
IR	0.26	0.70	0.38	103
JK	0.02	0.83	0.05	12
NF	0.16	0.31	0.21	54
O	0.01	0.50	0.01	4
OQ	0.92	0.95	0.93	226
PA	0.69	0.85	0.76	364
PF	0.29	0.75	0.41	87
RQ	0.20	0.49	0.28	51
micro avg	0.31	0.67	0.42	1257
macro avg	0.29	0.62	0.34	1257
weighted avg	0.51	0.67	0.56	1257
samples avg	0.43	0.71	0.49	1257

2.1.5 ML-KNN Results

Validation Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	66

FD	0.43	0.21	0.28	193
FQ	0.60	0.05	0.09	65
GG	0.62	0.24	0.35	41
IR	0.41	0.11	0.17	84
JK	0.20	0.08	0.12	12
NF	0.00	0.00	0.00	47
O	0.00	0.00	0.00	2
OQ	0.96	0.87	0.91	221
PA	0.66	0.85	0.74	365
PF	0.71	0.51	0.59	97
RQ	0.00	0.00	0.00	49
micro avg	0.69	0.50	0.58	1242
macro avg	0.38	0.24	0.27	1242
weighted avg	0.57	0.50	0.50	1242
samples avg	0.60	0.55	0.56	1242

Test Classification Report:

	precision	recall	f1-score	support
CQ	0.00	0.00	0.00	46
FD	0.46	0.19	0.27	226
FQ	0.00	0.00	0.00	59
GG	0.40	0.24	0.30	25
IR	0.40	0.08	0.13	103
JK	0.17	0.08	0.11	12
NF	0.00	0.00	0.00	54
O	0.00	0.00	0.00	4
OQ	0.95	0.85	0.90	226
PA	0.64	0.87	0.74	364
PF	0.63	0.38	0.47	87
RQ	0.00	0.00	0.00	51
micro avg	0.68	0.48	0.56	1257
macro avg	0.30	0.22	0.24	1257
weighted avg	0.53	0.48	0.47	1257
samples avg	0.59	0.53	0.54	1257

2.1.6 Comparison with Baseline Results from the Original Paper

Table 3 summarizes the accuracy, precision, recall, and F1 scores of the baseline classifiers as reported in the original paper [1]. Compared to our reproduced results, we observe that the overall trend of model performance is consistent, with SVM, Random Forest, and AdaBoost generally outperforming ML-kNN and Naive Bayes.

Table 3: Experiment results for baseline classifiers (from [1])

Methods	**Acc**	**Precision**	**Recall**	**F1**
ML-kNN	0.4715	0.6322	0.4471	0.5238
NaiveBayes	0.4870	0.5563	0.4988	0.5260
SVM	0.6342	0.7270	0.5847	0.6481
RandForest	0.6268	**0.7657**	0.5903	**0.6667**
AdaBoost	**0.6399**	0.7247	**0.6030**	0.6583

However, there are some noticeable differences in the exact values:

- Our Random Forest model achieved a test F1-score of **0.60**, compared to the paper's **0.6667**. Although the precision remains similar, our recall is relatively lower, possibly due to preprocessing or feature imbalance.
- AdaBoost in our setting yields a test F1-score of **0.61**, closely matching the paper's **0.6583**, and even surpasses it slightly in recall.
- ML-kNN yields the lowest F1-score among our reproduced models. This suggests that the weaker performance may be inherent to the method itself in this task setting, rather than caused by implementation issues.

In general, our reproduction confirms the paper's conclusion: ensemble-based models such as Random Forest and AdaBoost provide more stable and accurate classification across intent categories.

2.2 Feature Group Evaluation

Following the methodology of the original paper [1], we divided the handcrafted features into three categories: **Structure features, Content features and Sentiment features**.

To evaluate the importance of each group, we conducted a series of ablation experiments using Random Forest classifiers. We trained models using each feature group individually, and then in pairwise combinations, as well as all three together. The results are summarized in Table 4.

Table 4: Micro- and Macro-F1 scores using different combinations of feature groups

Feature Group(s)	**Micro-F1**	**Macro-F1**
Content	0.2898	0.1666
Structural	0.5696	0.2557
Sentiment	0.2721	0.1484
Content + Structural	0.6187	0.2927
Content + Sentiment	0.3723	0.2019
Structural + Sentiment	0.6278	0.3035
All (Full Set)	**0.6495**	**0.3331**

These results closely mirror the original findings. Structural features play a dominant role in classification performance.

2.3 Feature Importance Analysis

To further understand the relative impact of each handcrafted feature, we conducted a feature importance analysis using a Random Forest classifier within a Label Powerset (LP) multi-label classification framework.

Top features according to normalized importance (scaled to max = 1.0) are shown in Table 5.

We observe that content similarity features such as `init_sim` and `thread_sim`, as well as structural features like `abs_pos` and `norm_pos`, play dominant roles in model prediction. Sentiment features like `sent_pos` and `sent_neu` also show moderate influence.

Lower-ranked features include question words (e.g., `who`, `when`, `where`) and binary cues (e.g., `duplicate`, `has_thanks`), indicating these may be less informative in isolation. This result aligns well with our findings in Section 2.2, further confirming the relative strength of structural features.

Interestingly, our results show that sentiment features such as `sent_pos`, `sent_neu`, and `sent_neg` rank relatively higher in importance compared to what was reported in the original paper [1]. This difference might stem from the specific sentiment lexicons or polarity scoring tools we used, which may better capture emotional nuances in our dataset.

Apart from this deviation, the overall ranking of feature importance aligns well with the original findings. Structural features still dominate the top positions, reaffirming their central role in user intent prediction.

Table 5: Top features ranked by normalized importance (max = 1.0)

Rank	Feature Name	Normalized Importance
1	init_sim	1.0000
2	abs_pos	0.9399
3	thread_sim	0.7594
4	norm_pos	0.7006
5	sent_pos	0.6578
6	sent_neu	0.6480
7	length	0.5818
8	unique_len	0.5225
9	is_starter	0.5221
10	stemmed_len	0.5098
11	sent_neg	0.4044
12	question_mark	0.2206
13	has_thanks	0.1446
14	who	0.1158
15	what	0.1044
16	exclam_mark	0.1039
17	duplicate	0.0916
18	how	0.0887
19	lexicon_pos	0.0773
20	why	0.0534
21	verbal_feedback	0.0508
22	when	0.0464
23	lexicon_neg	0.0390
24	where	0.0349

3 Neural Models Reproduction

3.1 Introduction

This section aim to reproduce and evaluate the neural models proposed in the paper [1]. The task consists in classifying user utterances into 12 fine-grained intent categories based on their role in multi-turn conversations.

3.2 Reproduction Setup

The reproduction process followed these key steps:

1. Requested and obtained the MSDialog dataset.
2. Downloaded and inspected the provided codebase.
3. Set up a virtual environment and installed dependencies.

4. Trained word embeddings using the entire MSDialog corpus.
5. Ran all neural models locally and collected performance metrics.

3.3 Models Trained

The following neural models were successfully reproduced:

- BiLSTM
- BiLSTM-Context
- CNN
- CNN-MFS
- Char-CNN
- CNN-Feature
- CNN-Context
- CNN-Context-Rep

Table 6: Comparison of neural model performance (paper vs. reproduced)

Method	**Accuracy**	**Precision**	**Recall**	**F1**
BiLSTM (paper)	0.5515	0.6284	0.5274	0.5735
BiLSTM (ours)	0.5349	0.6310	0.5020	0.5591
CNN (paper)	0.6364	0.7152	0.6054	0.6558
CNN (ours)	0.5400	0.6320	0.4877	0.5505
CNN-MFS (paper)	0.6342	0.7308	0.5919	0.6541
CNN-MFS (ours)	0.5918	0.6900	0.5418	0.6070
Char-CNN (paper)	0.5419	0.6350	0.4940	0.5557
Char-CNN (ours)	0.3569	0.2689	0.1286	0.1532
BiLSTM-Context (paper)	0.6006	0.6951	0.5640	0.6227
BiLSTM-Context (ours)	0.5655	0.6697	0.5290	0.5911
CNN-Feature (paper)	0.6509	0.7619	0.6110	0.6781
CNN-Feature (ours)	0.5770	0.6667	0.5569	0.6068
CNN-Context (paper)	0.6555	0.7577	0.6070	0.6740
CNN-Context (ours)	0.6146	0.7223	0.5609	0.6314
CNN-Context-Rep (paper)	0.6885	0.7883	0.6516	0.7134
CNN-Context-Rep (ours)	0.6802	0.7887	0.6356	0.7040

3.4 Comparison with Paper Results

Although the reproduced models largely followed the original paper's structure and hyperparameters, some differences in performance were observed. The main causes include:

- **Training/Evaluation Variance:** Random seed differences and batch variations may have slightly affected results.
- **Pipeline Bugs:** Some preprocessing steps were incomplete or unclear in the original codebase.
- **Hyperparameter Deviations:** Although the reported hyperparameters were used, they were manually re-verified.
- **Missing Components:** Pipelines for CNN-MFS and Char-CNN were either buggy or missing and had to be reconstructed.

3.5 Attempted Improvements: CNN-Context-Rep

Efforts were made to improve the CNN-Context-Rep model. Enhancements included:

- Cleaner implementation using shared convolutional blocks for current, previous, and next segments.
- Accurate context slicing using lambda layers with explicit output shapes.
- Switched to Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and learning rate 1×10^{-3}.
- Training stabilization using early stopping with `restore_best_weights=True` and patience of 3 epochs.
- Fixed max sequence length to 800, filters to 1024, and dense layer size to 256.
- Added output thresholding: if no class probability exceeded 0.5, fallback to argmax.

The improved model achieved the following performance:

- **Accuracy:** 0.6218
- **Precision:** 0.7357
- **Recall:** 0.5712
- **F1-score:** 0.6431

Although this did not outperform the original CNN-Context-Rep in terms of F1-score, the new implementation is more modular and maintainable. With further tuning or feature fusion, additional gains could be achieved.

3.6 Conclusion

Overall, this reproduction validated the paper's core conclusion: incorporating conversational context improves model performance in user intent prediction. Despite the challenges of reproducing older codebases, the results remain consistent with the original findings. This work highlights the importance of clean and well-documented code for reproducibility in academic research.

4 Conclusion

In this project, we reproduced and evaluated both traditional and neural models for user intent prediction in information-seeking conversations, following the methodology described in the paper [1]. Our goal was to assess the replicability of the proposed approaches and gain insights into the effectiveness of various features and model architectures.

In the first part, we re-implemented five traditional machine learning classifiers using handcrafted features. We observed that ensemble-based models such as Random Forest and AdaBoost consistently outperformed other methods, and our results largely aligned with those reported in the original paper. Feature ablation and importance analysis further confirmed that structural and content-related features were most impactful, with sentiment features also contributing more than initially expected in our experiments.

In the second part, we reproduced several neural models, including BiLSTM, CNN variants, and context-aware architectures. While most models showed slightly lower performance than the original paper due to implementation variance or preprocessing gaps, the overall ranking and trends remained consistent. In particular, the CNN-Context-Rep model demonstrated strong performance, and our attempted improvements showed promise in terms of modularity and maintainability.

Through this reproduction, we confirmed the paper's core finding: incorporating conversational context significantly improves intent classification performance. Additionally, we gained practical experience in working with multi-label classification, dialogue datasets, feature engineering, and deep learning model tuning. This project also highlighted the importance of reproducible code and detailed documentation for advancing research reliability in the NLP and IR communities.

References

[1] Chen Qu, Liu Yang, W Bruce Croft, Yongfeng Zhang, Johanne R Trippas, and Minghui Qiu. User intent prediction in information-seeking conversations. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*, pages 25–33, 2019.