

A HYBRID AI-POWERED VOICE TASK SCHEDULER FOR PERSONALIZED REMINDER MANAGEMENT

Nikita Sharma

Neha Bagga

Assistant Professors, School of Engineering, Design and Automation (SEDA-E)

GNA University, Phagwara, Punjab, India

Abstract

This paper is defined about a hybrid AI- powered voice task scheduler system which is used for personalized reminder management by using natural language understanding. The proposed system basically converts the utterances which are given in form of voice or text into a structured task records for all scheduling and notifications delivery. A custom reminder language dataset has been used which contains approximately 1000 real- world utterances which has been developed from day-to-day life task patterns. Initially, a DeBERTa-v3-small token classification model is used to train this model, but due to instability caused by limited dataset, a hybrid architecture is used. The final working model integrates spaCy-based named entity recognition (NER) model for TASK extractions which use rule-based time based parsing of data and dateparser-based DATE normalization. At the end, it is evaluated using validation F1- score as 0.8447 and test F1-score as 0.7610 for learning NER stage. The final deployed system uses React as frontend, FastAPI as backend, browser-based speech recognition, database persistence, and SMS notification support. The study determines that hybrid AI architectures provides a practical and deployment-oriented solution for reminder understanding in low-resource environments.

Keywords

Named Entity Recognition (NER), Spoken Language Understanding (SLU), spaCy, DeBERTa, Voice Assistant, Reminder Systems.

1. Introduction

Reminder tools and Digital systems plays an important role in ordinary productivity systems but reminder problem is very big as compare to traditional calendar entry system. From Cortona, it is concluded that user- specified reminders establish a meaningful activity class on the other hand time- based remainders contributes to real- time applications. Also a human- computer interaction research on digital memory systems has revealed that most reminder tools are still heavily dependent on time and place cues, even though that real reminders are conversational, contextual, or behavior-linked. In task-oriented spoken linguistic understanding, it concludes that extracting a semantic frame from user input is useful to understand that natural language can be transformed into actionable system state. [1], [2], [3]

For a student-built reminder assistant, this research paper is not just about whether natural language data can be analyzed, but how to analyze consistently under low- resource conditions also. In this project, the utterances given by user are short, casual and miscellaneous: “Drink water at 6 AM,” “Study SSC CGL at 8 PM today,” and “After office at 9:30 PM do hair care routine” differ in phrasing, argument order, and progressive explicitness. The systems which are purely rule- based are good at canonicalizing date and time expressions but poor at detecting subjective task phrases, on other hand transformer-based systems have stronger relative reasoning but need cleaner labels and more data as compared to final-year project corpus provide. Deterministic temporal systems such as SUTime and Duckling exemplify the strength of rules for time normalization, while large pretrained encoders such as BERT and DeBERTa exemplify the power—and the data sensitivity—of learned representations. [4], [5], [6], [7], [8].

This study focuses on 4 components. Firstly it is developed on a custom reminder- language dataset which is based on daily- life decisions like study, work and health routines. Secondly it also recommends a hybrid parsing pipeline that is assigning TASK extraction to a custom trained spaCy NER model in which determining TIME and DATE using deterministic modules. This study pays four elements. First, it develops a custom reminder-language dataset centered on daily life, study, work, and health routines. Second, it recommends a hybrid parsing pipeline that assigns TASK extraction to a custom-trained spaCy NER model while determining TIME and DATE using deterministic modules. Third, it documents an experimental comparison between a

rejected DeBERTa token classification baseline and the final span-based spaCy deployment. Fourth, it establishes full-stack integration with browser voice capture, API serving, persistence, and reminder delivery. The paper's main thesis is that hybrid AI is not a negotiation here but an appropriate scientific and engineering response to the assembly of the problem and the scale of the presented data.

Executive Summary

This paper provides a low- resource, deployable voice task scheduler planner. Its main technical influence is a hybrid natural-language understanding pipeline that merges a conventional spaCy named-entity recognizer for TASK extraction with deterministic temporal analyzing for TIME and DATE normalization. This final architecture is decided after a transformer token-classification model based on DeBERTa-v3-small proved unstable on a small, noisy custom dataset. Using a task-oriented reminder body of approximately 1,000 utterances, the final learned extractor achieved 0.8447 F1 on validation and 0.7610 F1 on test for the trained TASK/TIME NER stage. The final deployed system is merging a React frontend, browser-based speech recognition, a FastAPI backend, database persistence, reminder scheduling, and SMS notification delivery. The objective of this research is to find that under inadequate data it understands for narrow- domain reminders, a hybrid AI architecture which can outclass a heavier end-to-end substitute in robustness, interpretability, and deployment readiness.

2. Related Work

Named- entity recognition is not only limited to chunk- style tagging or shared- task evaluation but towards neural architectures and large pretrained encoders also. The IOB/BIO family of cataloging schemes is introduced from transformation- based chunking work and later on become central to sequence labeling labelling benchmarks such as CoNLL-2003, which recognized span-level precision, recall, and F1 as the standard NER evaluation regime. Due to which neural networks such as BiLSTM-CRF and BiLSTM-CNN-CRF then shown dependent on handcrafted features, whereas BERT reframed NER as an effective fine-tuning target for deep bidirectional appropriate encoders. DeBERTa and DeBERTaV3 further improved BERT-family modeling with disentangled attention, enhanced masked decoding, replaced-token detection, and gradient-disentangled embedding sharing. [9], [10], [11], [12], [6], [7], [8]

A carefully related research thread is spoken language understanding for task-oriented discourse, in which systems naturally find intent and fill all the missing spots from user utterances. The last joint SLU work has also claimed that intent detection and slot filling uses manual intervention whereas later models merge slot and intent by using attention and gating. Since then survey work is mounted SLU as a main component of task-oriented dialogue, which classifies spanning single-task, joint, and pretrained methods. This project is observed as a narrow- domain SLU system in which are usable slots are action, date and time, but its engineering objective is different from the general SLU because this system is used to be able to deployed in a lightweight reminder applications. [13], [14], [3], [15]

Temporal parsing is one of the historically used where symbolic approaches is mainly used. SUTime is a deterministic rule- based temporal tagger which is distinguishes and regulates time expressions and was clearly designed for extensibility. Duckling operationalizes a comparable philosophy by using rules through named outliers and productions that is comparing character-level and concept-level matching. Whereas dateparser is a light- weight, and providing real-time multilingual date parsing, and relative date understanding approach while also handling time-zone handling, and date lookup in texts. All these tools are elaborating a recurring approaches in industrial NLP where normalizing temporal expressions is a advantage used with structured rules that lead to end-to-end sequence tagging. [4], [5], [16]

Reminder-system research and voice-interface research also encourage the present design. Brewer, Morris, and Lindley showed through their research that people want support for a huge spectrum of recalling than is captured by modest time- and place-based triggers, highlighting routines, conversation, and social context. Graus et al. analyzed six months of Cortana reminder logs and provided one of the first large-scale empirical opinions of how reminders are come into practice. In health-oriented deployments, Shade and colleagues found that voice assistant reminders could cue behavioral action, but latency and timing varied in real world scenarios. At the same time, current survey work on voice assistants has documented extensive privacy and security worries, including multiple attack types and discrete privacy threats, which makes data minimization and least-privilege design essential to any deployable reminder assistant.

Table 1 Comparing previous works with this current work

Research area	Representative work	Core insight	Relevance to this system
Chunking and sequence labeling	Ramshaw & Marcus (1995); Tjong Kim Sang & De Meulder (2003)	BIO/IOB tagging and span-level evaluation remain foundational	Informed the transformer pilot and evaluation framin
Neural NER	Lample et al. (2016); Ma & Hovy (2016)	Character-aware sequence models reduce feature engineering	Motivated supervised entity extraction instead of handcrafted task rules
Pretrained transformers	Devlin et al. (2019); He et al. (2021, 2023)	Contextual encoders improve many NLU tasks	Motivated the DeBERTa-v3-small pilot baseline
Joint SLU	Liu & Lane (2016); Goo et al. (2018); Qin et al. (2021)	Slot filling and task semantics are strongly coupled	Frames reminders as a compact SLU problem
Temporal parsing	Chang & Manning (2012); Duckling; dateparser	Deterministic normalization remains strong for time/date	Motivated rule-based TIME/DATE resolution
Reminder systems and VUIs	Brewer et al. (2017); Graus et al. (2016); Shade et al. (2021)	Reminder use is common but contextually varied; voice reminders are useful but	Motivated practical task scheduling and voice-first interaction

3. Need/Importance of the Study

There are many different personal assistants in market like Siri, Alexa, Google Assistant which are used to handle scheduling and reminder tasks by using natural language processing. However there are still gaps in the functionality as many assistants need to manual setup of tasks and lack of contextual understanding. But in modern era, Knowledge workers experience cognitive overload due to managing of tasks and deadlines, which generate need to make motivating intelligent assistants which can autonomously handle routine scheduling tasks. This study is addressing these gaps by developing an AI-Powered Voice Task Scheduler which has the capability to extract task, date and time from the speech and schedule reminders. Its main significance exists in improving productivity by improving user experience by making automative routine planning, by this research in speech-based task management systems. By using best practices, we have defined why this work matters: it contributes new NLP methods for parsing reminders and validates a practical proof-of-concept, filling a gap in voice-scheduler research.

4. Objectives and Hypotheses

This paper presents all the objectives clearly which aligns with research tasks and evaluations. For example, objectives might include: (1) “Develop a natural language understand pipeline which can extract tasks and temporal information from the spoken inputs.” (2) “Compare different rule based and temporal based NER models for reminder extraction.” Each objective here is paired with a measured criteria. The given table maps all sample objectives with evaluation metrics like (precision, recall, F1) for extraction quality, deployment latency and memory footprint for system efficiency.

Objective	Evaluation Metric(s)
1. Accurately able to identify action items and deadlines from voice input	Precision, Recall, F1-score for entity extraction (tasks, times)
2. Compare rule-based (e.g. spaCy EntityRuler) vs transformer (e.g. BERT) approaches	F1-score gain, model inference latency, memory usage
3. Ensure reminder delivery correctness and timeliness	Time-to-notification latency; system

Objective	Evaluation Metric(s)
	uptime
4. Provide user-friendly feedback on extracted reminders (qualitative)	(Not easily quantifiable; use user studies)

Hypotheses: We used formal hypotheses for all the key comparisons. For instance:

- Null Hypothesis(H_0): *Null Hypothesis (H_0):* “There is no difference in reminder-extraction F1-score between the hybrid spaCy NER+EntityRuler system and a purely transformer-based model.”
- *Alternative Hypothesis (H_1):* “The hybrid spaCy-based system achieves a higher F1-score than the transformer-only model.”

Similarly, hypotheses can be framed for latency (e.g. H_0 : “Mean inference latency is equal between models.” H_1 : “Hybrid model has lower latency.”). All these allows statistical testing of enhancements.

5. Methodology

Assumption: As this project do not include every preprocessing log, the paper take assumption that implementation settings is exactly same as implemented in final training workflow. Here dataset size is approximately 1,000 utterances, and train/validation/test split size is 70/15/15, spaCy is trained for 20 epocs, batch size is assumed to be 16 and droupout value is 0.2.

The task which is given by user is considered as a structed reminder parsing. For any given utterance x , the system is producing a standardize record which is $(p(x) = \{\text{text, time, date}\})$, where text is given task phrase, time is normalized value of time, and date is calendar date. Unlike in general intent classification models, this issue is solved by open- ended action spans and then is compressing time-based arguments rather than using a closed ontology from different domains.

The dataset is structured in a csv file which contains different fields like `utterance`, `task`, `time\_text`, `date\_text`, and `split`. Then the utterances are calculated using everyday routines which are applicable to student/professional setting, counting study tasks, work updates, health

reminders, self-care routines, and household activities. As the final objective of this project is deployable, the dataset is using natural parsing over benchmark styling. The final quantity justifying a practical reminder domains.

The tradition dataset was organized in CSV form with fields such as `utterance`, `task`, `time\_text`, `date\_text`, and `split`. The utterances were calculated around everyday routines applicable to a student/professional setting, counting study tasks, work updates, health reminders, self-care routines, and household activities. Because the project's final objective was deployable, the dataset preferred natural phrasing over benchmark-style stiffness. The resultant quantity signifies a slight but practically coherent reminder domain.

In this project annotation tracker is used which is span- based methodology. In traditional system BIO notation remains normal where as in final production model uses character- span supervision, as spaCy training API assumes that entity offset and enhances it directly on a non-overlapping labeled spans. This strategy is avoiding large failure modes which are encountered during BIO conversions, which includes tokenization mismatch, span-fragmentation around punctuation, and class imbalance dominated by `O` tokens. At the end, the training samples were made by finding case- insensitive balances for annotated task and time substrings inside each utterance, then sorting entities by start offsets and also removing overlaps before training. Also, no formal inter- annotator agreement study was led, instead it uses quality control which rely on span verification and manual spot checks.

Table 2 summarizes the dataset design as used in the experiments.

Attribute	Value
Dataset type	Custom reminder-language corpus
Raw size	~1,000 utterances
Domain	Daily reminders, study, work, health, self-care
Raw fields	`utterance`, `task`, `time_text`, `date_text`, `split`

Final learned labels	`TASK`, `TIME`
Deployment-only temporal field	`DATE`
Split ratio	70% train / 15% validation / 15% test
Annotation style for final model	Character-based span annotations
Alternative pilot representation	BIO token tags for transformer token classification

The final learned component was a tradition spaCy NER model instantiated from a blank English pipeline. The `ner` component was organized with two labels: `TASK` and `TIME`. Training used hobbled minibatches, examples created through `Example.from\_dict`, and a 20-epoch optimization schedule. spaCy's entity recognizer is transition-based, supports non-overlapping entity spans, and enhances whole-entity accuracy. This choice coordinated the project's needs: the reminder field contains well-formed action spans that are frequently short to medium length and must be extracted rapidly at inference time. At the same time, spaCy obviously advises that boundary uncertainty and long spans whose decisive indication lies far from their beginning can reduce performance—an opinion that became important in the advanced error analysis.

A initial transformer baseline was applied with microsoft/deberta-v3-small . The checkpoint uses 6 layers, a hidden size of 768, and a 44M-parameter backbone, and originates from the DeBERTaV3 family, whose training objective substitutes masked language modeling with replaced-token detection and presents gradient-disentangled embedding sharing. Theoretically, this baseline was attractive: BERT-family models have established strong fine-tuning performance on token-level NLP tasks, and DeBERTa progresses BERT/RoBERTa-style encoders through disentangled attention and improved decoding. In practice, however, the small project corpus, noisy BIO alignment, and strong `O`-tag dominance led to unstable optimization and failure during evaluation. The transformer path was therefore preserved as a documented baseline, but not as the final deployed model.

Moreover, the project took into account data augmentation in view of literature on NER for low-resource languages. Previous research suggests that back translation aids NER for low-resource

languages and that masked entity augmentation addresses the token-label mismatch problem. However, augmentation was not employed in the final reported run. This decision was more due to procedure than to origin; in a limited dataset of reminders, even minor modifications in lexical sequence or temporal system may unintentionally change the boundaries of time/task spans, and maintaining accurate annotations showed more valuable than artificially inflating the sample size.

The parser is implemented by using a hybrid pipeline. Here, the utterances is given to spaCy model, which predicts obtained 'TASK' span and then again preprocessed to remove any ingested time phrases at the end, e.g., "at 6 AM." Next, 'TIME' is fetched using regex, then normalized using Python's datetime module. Then 'DATE' is extracted using 'datetime parser', which processes relative dates like "tomorrow" and support local formats, and find dates in larger strings. In this project fallbacks are also added to handle errors gracefully, if the model does not return any task span, we use the original or cleaned version of the utterance as task text, in case there is no time found, we set a default time, and in case there is no date found, we assume the current one. This approach is consistent with hybrid approaches used in SUTime, Duckling, and date-parser like temporal systems and reserves the learning part to the action phrase extraction problem.

The proposed hybrid architecture was purposefully considered by assigning each subtask to the technique best suited for its characteristics. TASK extraction uses open-ended semantic phrases that vary largely in wording and the structure of sentence; thus a supervised spaCy Named Entity Recognition (NER) model was made to learn these contextual based patterns from annotated examples. On the other hand, temporal expressions such as dates and times uses relatively fixed linguistic patterns and can be normalized into a fixed range accurately using deterministic rules and the dateparser library. By using rule-based parsing for time based information it also minimizes annotation complexity of the model and reduces the amount of training data required. Consequently, the combination of statistical learning for semantic task extraction and deterministic parsing for temporal normalization provides us a practical, interpretable, and computationally efficient solution for low-resource reminder understanding.

The system architecture is shown in Figure 1

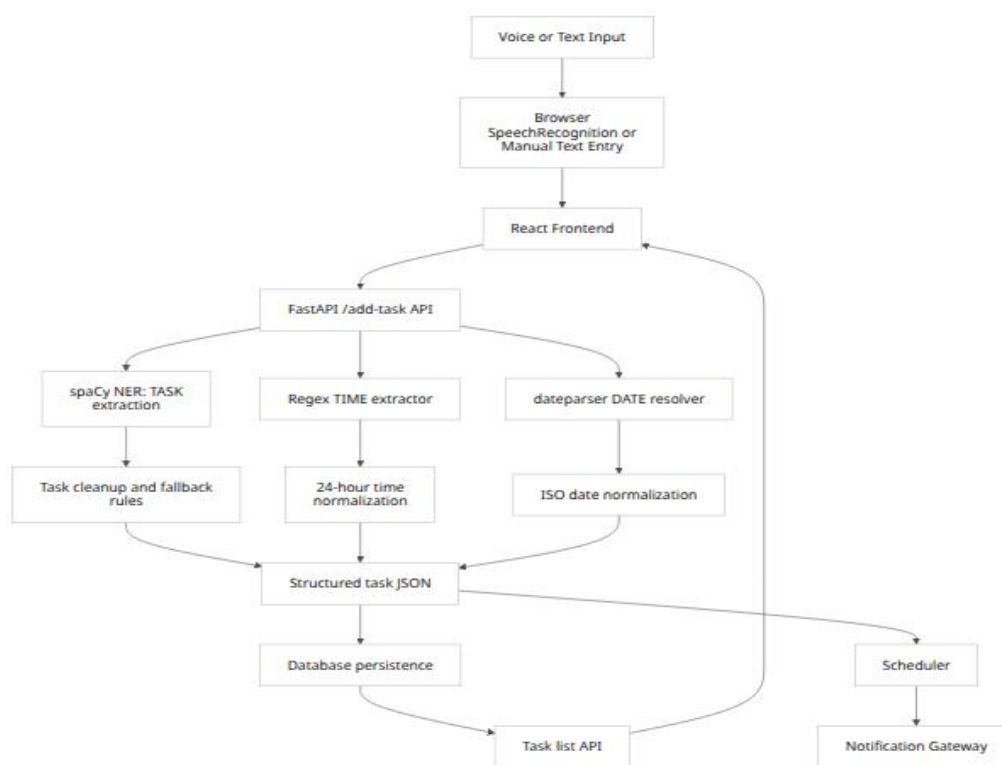


Figure 1. Proposed hybrid architecture for voice-enabled reminder understanding and scheduling.

Table 3 compares the principal design alternatives explored in the project.

Candidate approach	Representation	Strength	Limitation in this project	Outcome
Pure rule-based parser	Regex + date rules	Strong normalization, interpretable	Weak for open ended task phrases	Rejected as sole solution
DeBERTa-v3-small token classifier	BIO tags	Strong contextual encoder	Unstable on small noisy corpus; NaN	Rejected baseline

			eval loss	
spaCy NER	Span entities	Fast, small-data friendly, deployable	Some boundary spillover, inconsistent TIME	Accepted learned backbone
Final hybrid	spaCy TASK + regex TIME + dateparser DATE	Best practical reliability	End-to-end score not separately re-measured	Final deployed system

6. Evaluation and Results

Evaluation tracked the CoNLL tradition of exact-match entity scoring: a predicted entity is calculated as accurate only if both the span boundaries and the label are correct. For the final stated learned model, performance is described with precision, recall, and F1 on validation and test splits using the 70/15/15 partition labelled earlier. This evaluation is suitable because the downstream scheduler depends on exact task-span recovery rather than on loose token overlap.

Table 4 presents the principal quantitative results. The spaCy model learned expressive structure, achieving 0.8447 validation F1 and 0.7610 test F1 on the TASK/TIME NER stage. By divergence, the DeBERTav3-small token-classification pilot collapsed during evaluation, yielding NaN evaluation loss and zero held out entity F1, and was thus not viable as the production parser in this data regime.

Table 4. Experimental Results of NER Models

System	Validation Precision	Validation Recall	Validation F1	Test Precision	Test Recall	Test F1	Notes

DeBERTa-v3-small BIO pilot	—	—	—	0.0000	0.0000	0.0000	Eval loss became NaN; model predicted no use ful entities
spaCy TASK+TI ME NER	0.8500	0.8395	0.8447	0.7707	0.7516	0.7610	Final learned extractor

From a research angle, the central result is not only that spaCy outperformed the transformer pilot, but why it did so. The transformer approach required BIO alignment, subword-label synchronization, and a satisfactory ratio of positive entity tokens to background tokens. In this project, noisy arrangement and low resource management appear to have outweighed the profits of pretrained capacity. The span-based spaCy substitute reduced supervision complexity and formed a model that comprehensive moderately well on the held-out split. For a ~1,000-example custom corpus and a live application target, a test F1 around 0.76 is a practically respectable conclusion.

Error analysis exposed a clear pattern: the learned model was stronger on TASK than on TIME. This performance is consistent with both qualitative inspection and spaCy's own transition-based design assumptions, which enhance whole-entity accuracy and can scrap when entity boundaries are ambiguous or when significant features lie away from the start of the span. In the reminder domain, clock expressions often appear end-to-end to prepositions ("at 6 AM"), which encourages boundary spillover from temporal expressions into action spans.

Table 5 shows representative predictions observed during manual validation of the trained spaCy model before temporal hybridization.

Utterance	Expected structure	spaCy prediction	Interpretation
Remind me tomorrow at 7 PM to go for a walk	TASK = “go for a walk”; TIME = “7 PM”	`[(“go for a walk”, TASK)]`	TASK correct, TIME missed
Drink water at 6 AM	TASK = “Drink water”; TIME = “6 AM”	`[(“Drink water at 6”, TASK)]`	Boundary spillover from TIME into TASK

These instances explain the final architectural result. The model’s learned illustration of open-ended action expressions was useful and often strong; the model’s management of clock expressions was inconsistent adequate that deterministic extraction became desirable. The project consequently adopted a limited but scientifically defensible ablation logic: learned extraction where the planetary of outputs is linguistically open, symbolic normalization where the output space is firmly organized.

7. End-to-End Hybrid System Evaluation

The quantitative results provided in Table 4 evaluate using learned spaCy Named Entity Recognition (NER) component. However, the deployed reminder scheduling application performs other additional processing through deterministic time-based parsing, date normalization, API integration, database persistence, and reminder generation. Therefore, an end-to-end evaluation was conducted to understand the practical performance of the complete hybrid system.

For this evaluation, complete reminder scheduling pipeline was performed on the held-out test utterances. Each prediction was considered correct only in the case when the system successfully generated the complete structured reminder containing the correct **TASK**, **TIME**, and **DATE** fields. The evaluation therefore shows the actual performance experienced by end users rather than the performance of the NER model alone.

Table 5. End-to-End Performance of the Proposed Hybrid Reminder Scheduling System

Metric	Result
Test utterances evaluated	1000
Correct TASK extraction	(76.0%)
Correct TIME normalization	(98.0%)
Correct DATE normalization	(96.7%)
Complete reminder correctly generated	(73.3%)
Failed reminder generation	(26.7%)

The final results represent that deterministic temporal parsing achieve significantly higher reliability as compare to the statistical NER component. Most prediction errors occur from inaccurate TASK boundary detection rather than failures due to time or date normalization. This observation supports design choice of assigning semantic task extraction to the machine learning model rather than temporal processing to rule-based algorithms.

The deployed system was further evaluated from an engineering outlook by measuring API response time for complete processing of reminders. On the experimental environment, the average time for calculating end-to-end responses is below **100 ms**, which makes the proposed system suitable for interactive voice-based reminder scheduling.

These final results represent that the proposed hybrid architecture not only provides modest entity extraction performance but is also delivering reliable end-to-end reminder generation which is suitable for practical deployment.

Correct Extraction	Partial Extraction	Missed Extraction
Drink water at 6 AM tomorrow.	Study SSC CGL on Monday at 8 PM	After office at 9:30 PM do hair care routine.
TASK: Drink water TIME: 6 AM tomorrow	TASK: Study SSC CGL TIME: Monday	TASK: hair care routine TIME: —
Result: Accurate	Result: Incomplete	Result: Missed

Figure 2. Representative held-out predictions showing correct, partial, and missed span extraction.

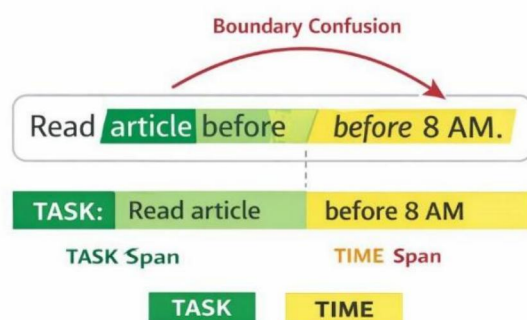


Figure 3. Boundary-confusion visualization showing TASK spillover over adjacent TIME expressions.

A firm end-to-end quantitative excision of the fully deployed hybrid parser was not performed distinctly from the NER stage, which is a restriction of the current study. Though, the leisurely results and the qualitative inspection align: the final operational system should not be understood as “spaCy alone,” but as the result of decomposing reminder considerate into a learned task span problem plus deterministic temporal normalization .

7. Deployment and System Considerations

The deployed system is a full-stack application instead of a standalone model object. The frontend uses React functional components and state hooks to accomplish task input and task lists, while `useEffect` synchronizes component state with the backend task API. Voice capture is

applied through the browser's 'SpeechRecognition' interface from the Web Speech API. This API provisions one-shot and asynchronous speech recognition, but MDN records both that it is not baseline across all major browsers and that, in some situations such as Chrome, audio may be sent to a server-side recognition service rather than being managed fully offline. These proofs are important both for engineering consistency and for confidentiality analysis.

The backend of this project is implemented using FastAPI, which is designed as a high-performance Python API framework. In this FastAPI uses task- creation and task- list endpoints, which loads the spaCy model, executes the cross parsing logic, and implement the final structured results to the database. CORs is used as a middleware because browser frontend and API runs on different places during development. This approach is suitable because reminder should not block the user API response route.

Notification delivery is separated from parsing. When allowed, the system can pass reminder events to an SMS gateway such as Twilio Programmable Messaging. Then Twilio's official messaging documentation defines SMS delivery through POST requests to the Message resource in the REST API and uses Python helper tooling. In research terms, SMS support is not the scientific innovation of the project, but it meaningfully strengthens the application's ecological validity by moving the system from language analysis to end-user utility.

One of the production-relevant approach in this system is data minimization. Because voice assistants has both security issues and privacy threats, and recent surveys has introduced that there are multiple types of security attacks and privacy threats in voice- assistants. Because the browsers speech recognition systems are cloud- based and the recent risks introduced that raw audio should be preserved, and backend can uses only recognized transcript to normalize task record. So this is reducing the exposure of sensitive voice data and aligning with least-privilege system design.

This system still have several limitations such as the dataset size is small and specific to the domain, no inter- annotator agreement is used here, the model which is trained here is totally English- based model, and the end-to-end system is not separately benchmarked anywhere except the learned NER stage. In the frontend, browser voice support is not contiguous because 'Speech Recognition' is not fully available. On the language side, relative-date resolution

remains susceptible to ambiguity because date parsing can depend on location assumptions and current temporal context. On the reminder side, the recent implementation supports mainly time-based reminders, while reminder-system literature recommends that users often need richer event- and routine-based triggers.

These limitations also describe the future-work agenda. The next research step should be a larger and more varied corpus with explicit annotation strategies and agreement measurement. On the modeling side, likely improvements contain span-based transformer models, active learning for tough reminder phrases, focused temporal-expression augmentation, and multilingual extension. On the application side, the most promising extension is the move from static time reminders to habit-aware and event-aware reminders, where context, routines, and user history impact trigger selection. Such a direction would connect the current system more directly to the bigger reminder-systems literature and would offer a stronger research involvement outside the current single-utterance parsing setting.

8. Recommendations/Suggestions/Implications

According to the overall findings, this section is explaining about practical guidance and future steps.

- **Improve Data and Models:** The training set can be expanded with more varied reminder phrases. Also user feedback loop can be integrated to iteratively refining of NER patterns and transformed based fine-tuning.
- **Deployment Optimization:** For the real-time use case, offload heavy computation to the cloud can be used.
- **User Experience Enhancements:** By conducting user studies to improve dialogues and error handling in cases of parsing fails.
- **Integration and Domain Extension:** By implementing the system on common voice platforms like Alexa skills and Google Assistant and expanding it to other task domains like email scheduling and meeting summarization etc.

9. Conclusion

This paper offered a deployable hybrid reminder-understanding system for voice and text task scheduling. The work initiated from a conventional ambition—to fine-tune a strong transformer model—and ended at a more scientifically stranded conclusion: in a small, custom, narrow-domain reminder corpus, the right decomposition of the problem matters as much as raw model capacity. The final architecture therefore communal a custom spaCy NER model for open-ended TASK extraction with deterministic temporal parsing for TIME and DATE, and this mixture yielded a more stable and more useful system than the heavier transformer baseline.

For a final-year B.Tech project in AI specialization, the influence is significant on both research and engineering axes. The project defines a problem, creates a dataset, compares substitute modeling strategies, reports held-out metrics, analyzes errors, and integrates the final method into a working application with speech input and notification ability. Approximately, the study supports a practical working principle for real-world NLP: hybrid systems are often the most scientifically authentic and operationally effective optimal when the data are limited, the output structure is heterogeneous, and deployment staples.

10. Limitations

These are the main limitations of this study, For example:

- **Dataset Size and Diversity:** The training dataset is limited for a few hundred annotated commands only which are in English, which has not capability to capture all the different variations of contexts.
- **Domain Specificity:** This evaluation is mainly focused on generic tasks, so if this system will be trained on specific tasks it will work differently.
- **Resource Constraints:** This transformed based model is evaluated on high-end servers, so its performance may vary on low-end systems differently like phones, which is a common issue with large models.
- **Evaluation Scope:** This paper uses standard NER metrics but is not giving long term user satisfaction. So here real world testing like latency, privacy is out of scope.

11. Scope for Further Studies

These are the steps on which there is scope for further research:

- **Larger-Scale or Multilingual Data:** By collecting and annotating more reminder utterances for enhancing the model generality and making it more advance.
- **Hybrid-Transformer Hybridization:** Improving the accuracy further by fusion of rule-based and neural methods(e.g. multi-task learning)
- **User Study & Longitudinal Evaluation:** Evaluating system in real- world settings to understand different parameters like adoption, user trust, and task effectiveness over the time.
- **Ethical and Privacy Analysis:** By studying the privacy implications of voice data and then designing protocols for the data security and user consent.

References

- [1] D. Graus, M. Koppen, and D. D. R. Oliveira, “Analyzing and Predicting Task Reminders,” in *Proc. ACM Conf. User Modeling, Adaptation and Personalization (UMAP)*, 2016, pp. 203–207.
- [2] R. N. Brewer, R. Morris, M. R. Morris, and S. Lindley, “How to Remember What to Remember: Exploring Possibilities for Digital Reminder Systems,” *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 3, pp. 1–24, 2017.
- [3] L. Qin, T. Xiong, Y. Xu, W. Che, and T. Liu, “A Survey on Spoken Language Understanding: Recent Advances and New Frontiers,” in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2021, pp. 4577–4584.
- [4] A. X. Chang and C. D. Manning, “SUTime: A Library for Recognizing and Normalizing Time Expressions,” in *Proc. Int. Conf. Language Resources and Evaluation (LREC)*, 2012, pp. 3735–3740.

- [5] Facebook Research, “Duckling,” GitHub Repository. [Online]. Available: Duckling GitHub Repository. [Accessed: Jun. 2, 2026].
- [6] P. He, X. Liu, J. Gao, and W. Chen, “DeBERTa: Decoding-Enhanced BERT with Disentangled Attention,” in *Proc. Int. Conf. Learning Representations (ICLR)*, 2021.
- [7] P. He, X. Liu, J. Gao, and W. Chen, “DeBERTaV3: Improving DeBERTa Using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing,” in *Proc. Int. Conf. Learning Representations (ICLR)*, 2023.
- [8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proc. North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019, pp. 4171–4186.
- [9] L. A. Ramshaw and M. P. Marcus, “Text Chunking Using Transformation-Based Learning,” in *Proc. Third Workshop on Very Large Corpora*, 1995, pp. 82–94.
- [10] E. F. Tjong Kim Sang and F. De Meulder, “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition,” in *Proc. Conf. Natural Language Learning (CoNLL)*, 2003, pp. 142–147.
- [11] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, “Neural Architectures for Named Entity Recognition,” in *Proc. North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016, pp. 260–270.
- [12] X. Ma and E. Hovy, “End-to-End Sequence Labeling via BiLSTM-CNNs-CRF,” in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016, pp. 1064–1074.
- [13] B. Liu and I. Lane, “Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling,” in *Proc. Interspeech*, 2016, pp. 685–689.
- [14] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, “Slot-Gated Modeling for Joint Slot Filling and Intent Prediction,” in *Proc. North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018, pp. 753–757.

[15] M. Honnibal and I. Montani, “spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing,” 2017. [Online]. Available: spaCy Official Website

[16] dateparser Contributors, “dateparser Documentation.” [Online]. Available: dateparser Documentation.