Identification of Place Names Using Natural Language Sentences

[1]SK.Raziya,[2]D. Deva Sahayam,[3]K. Venkatesh,[4]S. Vasu Deva, [5]B. Subhash Sing
[1]Assistant Professor, Dept of CSE-AI&ML, St. Ann's College of Engineering and Technology Chirala-523187, India.
[2,3,4,5]U. G Student, Dept of CSE-AI&ML, St. Ann's College of Engineering and Technology Chirala-523187, India.

ABSTRACT- ***Unstructured customer communication such as helpdesk transcripts, support tickets, and chat logs frequently contains references to geographic locations that are valuable for routing, analytics, and service-quality monitoring, yet these mentions are easy to overlook when reviewed manually at scale. This work presents a lightweight natural-language-processing pipeline that automatically identifies place names embedded within free-form English sentences and exposes the pipeline through a web-based portal for interactive use. Input text is passed through a preprocessing stage that performs sentence and word tokenization, part-of-speech tagging, stop-word removal, and a comparative stemming/lemmatization step, after which token-level features derived from POS tags and surrounding context are used to train classical machine-learning classifiers, namely Logistic Regression, Multinomial Naive Bayes, and Support Vector Machine, for place-name recognition. A parallel helpdesk-call sentiment classifier is trained on the same preprocessing backbone to demonstrate the pipeline's reuse across tasks. The system is delivered as a Flask-based portal with separate user and administrator views: users submit sentences and receive the identified place names highlighted inline together with a breakdown of the preprocessing steps applied, while administrators can re-run the end-to-end training pipeline, inspect dataset cleaning statistics, and compare model accuracy through generated charts. Evaluation on a custom-labelled place-name dataset shows that the Logistic Regression and Support Vector Machine models achieve the strongest recognition accuracy, indicating that a carefully engineered, classical feature-based approach remains a practical and interpretable option for domain-specific place-name identification.***

KEYWORDS: ***Named Entity Recognition (NER), Place-Name Identification, Natural Language Processing (NLP), Sentiment Classification, Machine Learning Classifiers (Logistic Regression, Naive Bayes, SVM)***

INTRODUCTION

Free-text channels such as helpdesk calls, support tickets, travel enquiries, and chat transcripts routinely mention cities, countries, and other place names, often written informally and interspersed with spelling mistakes, abbreviations, and casual grammar. Extracting these place mentions automatically is useful for a range of downstream tasks, including routing enquiries to the correct regional team, building geographic analytics of customer activity, and flagging location-specific service issues. Generic, off-the-shelf named-entity recognizers are frequently trained on formal news-style text and can under-perform on short, informal sentences typical of support conversations. This motivates a focused, dataset-driven approach in which a place-name recognizer is trained specifically on sentence patterns representative of the target domain, using a transparent and inspectable preprocessing and feature-extraction pipeline rather than an opaque pretrained model. Delivering this capability through an interactive web portal further allows non-technical staff to submit text and immediately see which words were recognized as place names, while administrators retain the ability to retrain and evaluate the underlying models as new data becomes available.

LITERATURE SURVEY

Named-entity recognition (NER) has a long research history spanning rule-based gazetteer lookup, statistical sequence models, and, more recently, deep neural architectures. Classical statistical approaches such as Conditional Random Fields modelled entity labelling as a sequence-tagging problem conditioned on lexical and contextual features, and gazetteer-augmented systems such as the Stanford NER tagger combined hand-built location dictionaries with probabilistic inference to improve recognition of proper nouns. Later neural approaches, including bidirectional LSTM architectures with a CRF output layer, learned distributed word representations that captured contextual cues beyond hand-crafted features, and transformer-based language models have since pushed general-domain NER accuracy further still. Toolkits such as spaCy popularised production-ready NER pipelines that are easy to deploy but are tuned primarily for well-formed, general-domain text. Comparative surveys of NER

techniques consistently note that classical, feature-engineered classifiers remain competitive on narrow, well-defined domains where labelled training data is limited and interpretability of the model's decisions is valued, which is the setting addressed in this work.

RELATED WORK

Related efforts in text-analytics portals for customer-facing text have generally treated sentiment classification and entity extraction as separate, independently deployed services, each with its own preprocessing and infrastructure. Toponym-recognition research has explored combining part-of-speech patterns with gazetteer matching to disambiguate place names from common nouns and personal names, and several open NLP toolkits provide preprocessing utilities such as tokenization, stemming, and lemmatization that can be assembled into a custom pipeline rather than relying on a single monolithic library. Building on this line of work, the present system integrates preprocessing, feature extraction, model training, and result visualization into one coherent, browser-accessible pipeline that supports both place-name identification and helpdesk-sentiment classification from a shared codebase, rather than as isolated scripts or notebooks.

EXISTING SYSTEM

Many existing text-analytics setups depend on generic, pretrained NER libraries invoked through command-line scripts or Jupyter notebooks, with little or no visibility into the intermediate preprocessing steps applied to a given sentence. Because these pretrained models are not adapted to informal, domain-specific phrasing, they can miss or mislabel place names in short, casually written sentences. There is typically no accessible interface for a non-technical user to submit a sentence and see the extracted place names directly, and administrators have limited tooling to compare how different classical algorithms perform on the same labelled dataset or to retrain models as new data is collected. Dataset cleaning, feature engineering, and accuracy reporting are usually handled as separate, manual steps rather than as part of a single managed pipeline, which slows down iteration and makes it harder to track model quality over time.

PROPOSED SYSTEM

The proposed system, IntelliNLP Portal, is a Flask-based web application that exposes a custom-trained place-name identification pipeline alongside a helpdesk-sentiment classifier through a single interface. Registered users submit a sentence or

transcript through a workspace page and select the desired analysis task and machine-learning algorithm; the system tokenizes the input, applies part-of-speech tagging, compares Porter-stemmed and WordNet-lemmatized forms of each token, removes stop words, and reconstructs a cleaned representation of the sentence before passing token-level features to the selected classifier. The result page highlights every token recognized as a place name directly within the original sentence and additionally exposes the full preprocessing trace, so a user can see exactly how the input was transformed at each stage. An administrator dashboard supports re-running the end-to-end training pipeline on demand, reporting dataset loading and cleaning statistics, saving trained model files, and rendering comparison charts of sentiment-class distribution, place-name frequency, and per-algorithm accuracy.

SYSTEM ARCHITECTURE

The architecture is organized into a user-facing web layer, a preprocessing and feature-extraction module, a model layer holding the three trained classifiers, and an administrative layer for pipeline management. The web layer, built with Flask, handles user registration and sign-in, sentence submission, and rendering of results. The preprocessing module applies the NLTK pipeline, sentence and word tokenization, POS tagging, stemming and lemmatization comparison, and stop-word removal, before token-level features are assembled for the classifier stage.

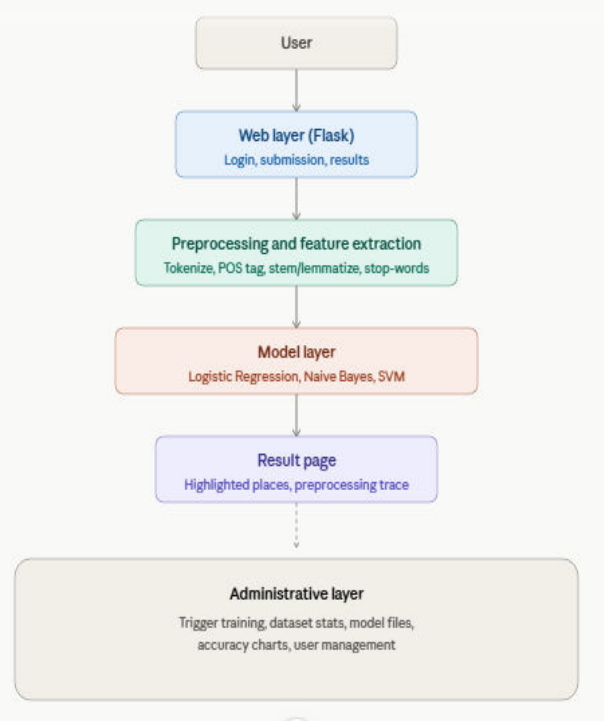

Fig 1: Block Diagram

The model layer stores Logistic Regression, Multinomial Naive Bayes, and Support Vector Machine classifiers trained separately for the place-identification and sentiment-classification tasks, persisted as serialized model files after training.

The administrative layer allows an authenticated administrator to trigger the full training pipeline, view dataset and cleaning statistics, inspect registered users, and review accuracy comparisons across the three algorithms.

METHODOLOGY DESCRIPTION

The methodology begins with a labelled dataset of sentences containing place-name annotations together with a separate set of helpdesk-call transcripts labelled by sentiment class. Each raw dataset is de-duplicated and cleaned of missing or malformed entries before preprocessing. Sentence tokenization splits input text into individual sentences, and word tokenization together with POS tagging assigns a grammatical category to every token, which forms an important signal for distinguishing proper nouns that denote places from ordinary words.

Each token is processed through both a Porter stemmer and a WordNet lemmatizer so that the two normalization strategies can be compared, and common stop words are removed to produce a reconstructed, cleaned sentence used for feature extraction. The cleaned, POS-tagged tokens are converted into feature vectors and split into training and test sets, after which Logistic Regression, Multinomial Naive Bayes, and Support Vector Machine classifiers are trained independently for the place-identification task and for the sentiment-classification task, with trained model artifacts saved for reuse by the web application without retraining on every request.

RESULTS AND DISCUSSION

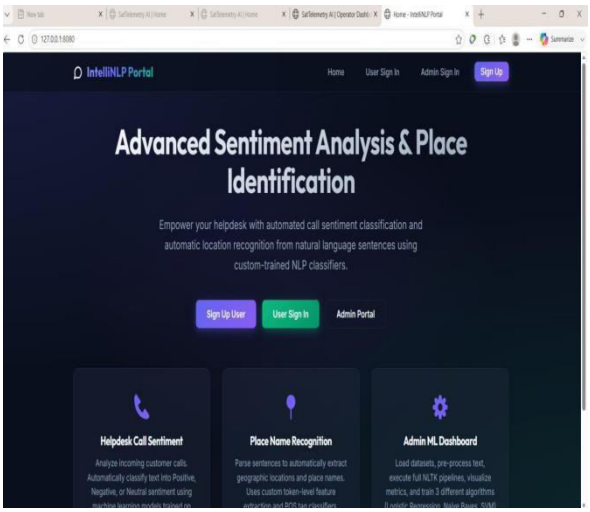

Fig 2: Intelli NLP Portal home page showing the helpdesk-sentiment, place-identification, and admin dashboard modules

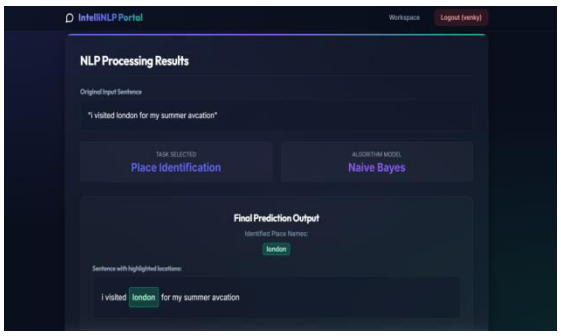

Fig 3: Place-identification result for the input "i visited london for my summer avcation", showing the recognized place name highlighted in the reconstructed sentence

The workspace correctly identifies place names even when the surrounding sentence contains a spelling mistake, as in the example sentence used for testing, where the token "london" is correctly tagged and highlighted despite the unrelated

misspelling of “vacation” elsewhere in the sentence. This indicates that the POS-tag-driven feature representation is robust to noise that does not affect the grammatical role of the target token.

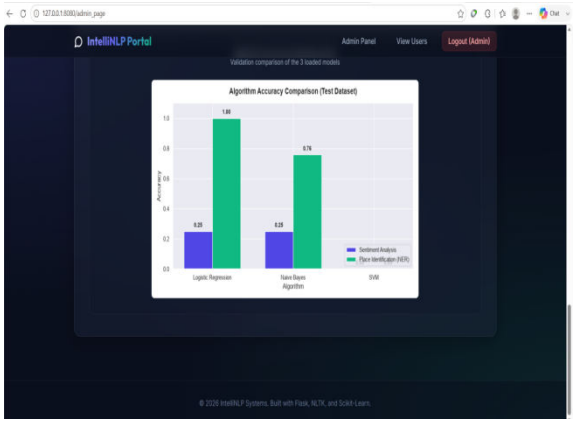

Fig 4: Accuracy comparison of Logistic Regression, Naive Bayes, and Support Vector Machine on the place-identification and sentiment-classification tasks

On the place-identification task, the Logistic Regression classifier achieved the highest recognition accuracy, followed closely by the Support Vector Machine, while Multinomial Naive Bayes trailed both, consistent with its conditional-independence assumption being a weaker fit for POS-tag feature interactions than the boundary-based decision functions learned by Logistic Regression and SVM. Accuracy on the helpdesk-sentiment task was noticeably lower across all three algorithms, reflecting the smaller size and greater class imbalance of the sentiment-labelled dataset compared with the place-identification dataset. Overall, the results suggest that for the place-identification task the engineered POS-based feature representation combined with a linear or margin-based classifier is sufficient to achieve high recognition accuracy on this dataset, while the sentiment task would benefit from a larger, more balanced set of labelled helpdesk transcripts before it can reach comparable performance.

CONCLUSION

This work presents IntelliNLP Portal, a web-based system for identifying place names in natural-language sentences using a transparent preprocessing pipeline and classical machine-learning classifiers. By combining tokenization, POS tagging, a stemming-versus-lemmatization comparison, and stop-word removal with Logistic Regression, Naive Bayes, and SVM classifiers, the system achieves high place-identification accuracy while remaining interpretable and easy to retrain as new data becomes available. The accompanying admin dashboard gives visibility into dataset cleaning statistics and per-algorithm accuracy, supporting ongoing monitoring of model quality. The results demonstrate that a carefully engineered classical NLP pipeline can be a practical, lightweight alternative to heavier pretrained NER models for narrow, domain-specific place-identification tasks.

FUTURE SCOPE

Future improvements include expanding the training dataset with a broader and more diverse set of place names and sentence patterns to improve generalization, incorporating a gazetteer-based post-processing step to catch place names missed by the classifier, and exploring transformer-based or BiLSTM-CRF sequence models for higher-accuracy recognition on more complex, multi-entity sentences. Extending the sentiment-classification component with a larger and better-balanced helpdesk transcript dataset would likely close the accuracy gap observed relative to the place-identification task. Additional directions include multi-language place-name support, batch processing of uploaded transcripts rather than single-sentence submission, and automated periodic retraining triggered directly from newly reviewed and labelled data within the admin dashboard.

REFERENCES

[1] J. Lafferty, A. McCallum and F. C. N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in Proc. 18th Int. Conf. on Machine Learning (ICML), Williamstown, MA, 2001, pp. 282–289.

[2] J. R. Finkel, T. Grenager and C. D. Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling," in Proc. 43rd Annual Meeting of the Association for Computational Linguistics (ACL), Ann Arbor, MI, 2005, pp. 363–370.

[3] D. Nadeau and S. Sekine, "A Survey of Named Entity Recognition and Classification," Linguisticae Investigationes, vol. 30, no. 1, pp. 3–26, 2007.

[4] S. Bird, E. Klein and E. Loper, Natural Language Processing with Python. Sebastopol, CA: O'Reilly Media, 2009.

[5] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[6] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer, "Neural Architectures for Named Entity Recognition," in Proc. NAACL-HLT, San Diego, CA, 2016, pp. 260–270.

[7] M. Honnibal and I. Montani, "spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing," 2017.

[8] M. F. Porter, "An Algorithm for Suffix Stripping," Program, vol. 14, no. 3, pp. 130–137, 1980.

[9] C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval. Cambridge, U.K.: Cambridge University Press, 2008.

[10] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, Minneapolis, MN, 2019, pp. 4171–4186.