NLP Final Year Project Ideas PDF

A good NLP final year project takes a real text problem, uses a public labelled dataset, compares a classic baseline such as TF-IDF with a fine-tuned transformer, and shows errors. Fake news detection, resume parsing with NER, question answering, summarisation and Hindi-English translation are strong choices.

Key takeaways

- Always compare a TF-IDF plus logistic regression baseline with a transformer.
- Hugging Face models and datasets make fine-tuning practical on free Colab GPUs.
- Use the task's standard metric: F1 for NER, ROUGE for summaries, BLEU or chrF for translation.
- Indian language projects stand out but need careful data choice.
- The PDF gives plans and steps, not finished source code.

Who this PDF is for

This PDF is for final-year B.Tech, MCA and M.Tech students working on text. The ideas cover classification, information extraction, question answering, generation and Indian languages. Each plan lists a dataset you can download, a Python stack (spaCy, scikit-learn, Hugging Face Transformers), modules for a team of 3 or 4, and viva talking points.

How to choose your project

- **Labelled data decides the project.** Check the dataset size and labels first.
- **Pick one task type**: classify, extract, answer or generate.
- **Plan compute.** Small models such as DistilBERT or MiniLM fine-tune on Colab.
- **Add an interface** (web form or chatbot) so examiners can type their own text.

Standard level projects

1. Fake News Detection

- **Problem:** Classify news articles as real or fake.
- **Dataset:** Kaggle "Fake and real news dataset"; LIAR dataset for short statements.
- **Tech stack:** Python, scikit-learn, Hugging Face Transformers, Streamlit.
- **Main modules:** Cleaning, TF-IDF baseline, fine-tuned BERT, explanation of key words, demo.
- **Steps:** Remove source-revealing text, train the baseline, fine-tune DistilBERT, compare F1 on both datasets.
- **Show in the viva:** Why the Kaggle set is easier than LIAR, and what that says about leakage.

- **Extensions:** Test on recent Indian news headlines you collect.

2. Toxic Comment Classifier

- **Problem:** Flag toxic, insulting or threatening comments.
- **Dataset:** Kaggle "Jigsaw Toxic Comment Classification Challenge".
- **Tech stack:** Python, PyTorch, Hugging Face, FastAPI.
- **Main modules:** Multi-label setup, class weights, transformer model, moderation API.
- **Steps:** Frame as multi-label, fine-tune a small transformer, tune thresholds per label.
- **Show in the viva:** Multi-label versus multi-class, and threshold choices.
- **Extensions:** Add a Hinglish test set you label yourself.

3. News Topic Classifier

- **Problem:** Sort news headlines into topics.
- **Dataset:** AG News dataset.
- **Tech stack:** Python, scikit-learn, Hugging Face.
- **Main modules:** Baseline, transformer, confusion matrix, batch classifier.
- **Steps:** Train TF-IDF with linear SVM, fine-tune DistilBERT, compare accuracy and speed.
- **Show in the viva:** When the simple model is good enough.
- **Extensions:** Add zero-shot labels for new topics.

4. Airline Tweet Sentiment Analysis

- **Problem:** Find customer sentiment and complaint reasons in tweets.
- **Dataset:** Kaggle "Twitter US Airline Sentiment".
- **Tech stack:** Python, NLTK or spaCy, Hugging Face, Plotly.
- **Main modules:** Tweet cleaning, sentiment model, reason classifier, dashboard.
- **Steps:** Clean handles and URLs, train sentiment and reason models, chart complaints by airline.
- **Show in the viva:** How emojis and sarcasm affect results.
- **Extensions:** Add aspect-based sentiment.

Advanced level projects

5. Resume Parser with Named Entity Recognition

- **Problem:** Extract name, skills, education and experience from resumes.
- **Dataset:** Train NER on CoNLL-2003 first, then label 100 to 200 resumes you collect with consent.
- **Tech stack:** Python, spaCy, a PDF text extractor, Flask.
- **Main modules:** PDF to text, custom NER, skill matcher, job match score.
- **Steps:** Extract text, label entities in a tool such as Doccano, train spaCy NER, match skills to a job description.
- **Show in the viva:** Entity-level precision, recall and F1.
- **Extensions:** Rank resumes against a job post.

6. Extractive and Abstractive Summarisation

- **Problem:** Summarise long news articles.

- **Dataset:** CNN/DailyMail dataset on Hugging Face.
- **Tech stack:** Python, Hugging Face (BART or T5), ROUGE.
- **Main modules:** TextRank baseline, fine-tuned or pretrained abstractive model, ROUGE scoring, UI.
- **Steps:** Run TextRank, run a pretrained summariser, compute ROUGE, compare samples.
- **Show in the viva:** Where abstractive summaries add facts that are not in the article.
- **Extensions:** Summarise Indian court judgments or college circulars.

7. Question Answering over Wikipedia Passages

- **Problem:** Answer questions from a given passage.
- **Dataset:** SQuAD 2.0.
- **Tech stack:** Python, Hugging Face, FastAPI.
- **Main modules:** Reader model, unanswerable-question handling, exact match and F1, chat UI.
- **Steps:** Fine-tune or use a pretrained QA model, test on SQuAD 2.0 dev, handle "no answer" cases.
- **Show in the viva:** Why SQuAD 2.0 adds unanswerable questions.
- **Extensions:** Add a retriever so it answers from your college handbook.

8. Hindi-English Machine Translation

- **Problem:** Translate sentences between Hindi and English.
- **Dataset:** IIT Bombay English-Hindi Parallel Corpus, or AI4Bharat Samanantar.
- **Tech stack:** Python, Hugging Face, a pretrained Indic translation model.
- **Main modules:** Data cleaning, baseline model, fine-tuning on a domain, BLEU and chrF scoring.
- **Steps:** Evaluate a pretrained model, fine-tune on a small domain set (for example, education notices), compare scores.
- **Show in the viva:** Why chrF suits Hindi better than BLEU alone.
- **Extensions:** Add speech input.

Research-oriented projects

9. Hate Speech Detection in Code-Mixed Text

- **Problem:** Detect hate speech in Hinglish social media posts.
- **Dataset:** Public code-mixed shared-task datasets (check the HASOC task pages for access).
- **Tech stack:** Python, Hugging Face, multilingual models such as XLM-R or MuRIL.
- **Main modules:** Transliteration handling, model comparison, error analysis.
- **Steps:** Compare mBERT, XLM-R and MuRIL, study errors on Roman-script Hindi.
- **Show in the viva:** Why code-mixing is hard for models.
- **Extensions:** Write up results for a student workshop.

10. Semantic Search for Research Papers

- **Problem:** Search papers by meaning, not exact words.
- **Dataset:** Kaggle "arXiv Dataset" (Cornell University) abstracts.
- **Tech stack:** Python, sentence-transformers, FAISS, Streamlit.

- **Main modules:** Embedding pipeline, vector index, keyword baseline (BM25), relevance tests.
- **Steps:** Embed abstracts, build a FAISS index, compare with BM25 on 30 test queries you judge.
- **Show in the viva:** Your relevance judgements and recall at 10.
- **Extensions:** Add a reranker.

11. Grammar Error Correction for Learners

- **Problem:** Correct grammar mistakes in English sentences.
- **Dataset:** Public grammar correction corpora such as the BEA-2019 shared task data (check licences).
- **Tech stack:** Python, Hugging Face (T5).
- **Main modules:** Sequence-to-sequence model, evaluation, feedback UI that shows edits.
- **Steps:** Fine-tune T5 on error-correction pairs, evaluate, display edits word by word.
- **Show in the viva:** How you measured correction quality.
- **Extensions:** Explain each correction in simple words.

12. Topic Modelling of Customer Feedback

- **Problem:** Find themes in large sets of reviews.
- **Dataset:** Amazon or app-store review dumps that are publicly licensed, or 20 Newsgroups for testing.
- **Tech stack:** Python, gensim (LDA), BERTopic.
- **Main modules:** Preprocessing, LDA, BERTopic, coherence scores, theme dashboard.
- **Steps:** Run LDA and BERTopic, compare coherence and human readability, label themes.
- **Show in the viva:** Why coherence scores and human judgement can disagree.
- **Extensions:** Track themes month by month.

Report structure

A final-year report is marked on clarity as much as on code. A structure most universities accept:

1. **Abstract** (150 to 250 words): problem, method, main result.
2. **Introduction:** background, problem statement, objectives and scope.
3. **Literature review:** 8 to 15 papers or existing systems, and the gap your project fills.
4. **Requirements and design:** functional and non-functional requirements, architecture diagram, the text pipeline from raw text to tokens to model to output.
5. **Implementation:** modules, tools, key algorithms and screenshots.
6. **Testing and results:** baseline and transformer scores on the standard metric, sample outputs and an error analysis.
7. **Conclusion and future scope.**
8. **References** in the format your department asks for (IEEE style is common).

Follow your own university's template if it gives one; it overrides this list.

Common mistakes to avoid

- Skipping the TF-IDF baseline, so gains cannot be judged.

- Leaving source names or URLs in fake news text, which leaks labels.
- Using generated text without checking it for made-up facts.
- Scraping social media against platform terms.

Frequently asked questions

Which NLP project is best for a final year?

Fake news detection, resume parsing with NER, summarisation and question answering are strong choices. They have public data and clear metrics.

Do I need to know transformers for an NLP project?

It helps. Most final-year NLP projects now fine-tune a small transformer such as DistilBERT, and compare it with a classic baseline.

Can I do an NLP project in Hindi or another Indian language?

Yes. Use datasets such as the IIT Bombay English-Hindi corpus or AI4Bharat resources, and multilingual models such as MuRIL or IndicBERT.

Which metric should I report?

F1 for classification and NER, exact match and F1 for QA, ROUGE for summaries, and BLEU or chrF for translation.

Does this PDF include full source code?

No. It gives the problem, data, stack, modules and steps for each project. You write the code yourself, which is what examiners test in the viva.

Is this PDF free?

Yes. Fill in the short form on this page and the PDF downloads right away.

Sources and further reading

- Hugging Face Datasets
- spaCy usage documentation
- SQuAD 2.0
- AI4Bharat
- Kaggle Datasets

About this PDF

Prepared by the Uncodemy training team as practice project ideas. Datasets and tools named are public; check each dataset's licence before use. We update it every January. Corrections: info@uncodemy.com.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550